南京大学

研 究 生 毕 业 论 文

（申请工程硕士学位）

论 文 题 目	基于机器学习的司法案例筛选系统设计与实现
作 者 姓 名	秦泽民
学科、专业名称	工程硕士(软件工程方向)
研 究 方 向	软件工程
指 导 教 师	刘嘉 副教授 何铁科 讲师

2019 年 4 月 11 日

学　　号：

论文答辩日期：**2019 年 5 月 08 日**

指 导 教 师：　　（签字）

The Design and Implementation of Judicial Case Screening System Based on Machine Learning

By

Zemin Qin

Supervised by

Associate Professor **Jia Liu** Lecturer **Tieke He**

A Thesis
Submitted to the Software Institute
and the Graduate School
of Nanjing University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Engineering

Software Institute

May 2019

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目： 基于机器学习的司法案例筛选系统设计与实现

工程硕士(软件工程方向) 专业 2019 级硕士生姓名： 秦泽民

指导教师（姓名、职称）： 刘嘉 副教授 何铁科 讲师

摘 要

司法文本逻辑严谨，结构清晰明确，适合采用计算机进行分析和处理。依托于近年来司法公开的相关政策，可用的司法语料大幅度增长，为机器学习方法在司法领域的应用提供了数据支撑。为了帮助司法从业者进行快速地案例检索，同时为民众法律询提供辅助工具，本文研究并设计实现司法案例筛选系统。这一系统的核心功能是对获取的文本进行分析，并从数据库内备选的海量裁判文书中筛选出相似案例供用户参考，同时包含裁判文书分析等辅助功能。

文章的主要工作是将机器学习及相关自然语言处理技术引入司法领域内。首先，本文对系统所属的司法智能领域进行了研究现状介绍。进一步，介绍了分词技术、文本关键词提取算法TF-IDF、文本分类技术fastText及web应用框架技术Django等与本文系统构建相关的技术，并完成系统设计及实现。

本文拟通过机器学习技术fastText对文本进行分类，进一步依据文本分类得到的标签结果进行相似案例筛选。系统的总体设计分为四部分，包括文件上传模块、文本输入模块、裁判文书分析模块和案例筛选模块。主要完成的工作如下：采用Django框架实现裁判文书上传模块和文本输入模块，用于实现与用户之间的交互，为整个系统的获取需要分析的文本。采用关键字匹配和TF-IDF算法抽取文本关键信息，实现对上传裁判文书的分析工作，帮助用户快速理解裁判文书。采用fastText框架训练多个机器学习模型，用于对输入文本进行分类；进一步实现依据分类标签及文本相似度的筛选和排序，最后展示筛选结果并提供下载等辅助功能。

司法案例筛选系统最终以web应用的形式运行良好。用户可以通过浏览器访问这一系统，进行文本输入、裁判文书上传和分析，并依据输入的文本进行相似案例的筛选。经实验，系统裁判文书解析正确率达到86%，案例筛选相关率达到81%,具有一定实用性。

关键词： 文本分类、案例筛选、TF-IDF算法、fastText框架

南京大学研究生毕业论文英文摘要首页用纸

THESIS: The Design and Implementation of Judicial Case Screening System Based on Machine Learning

SPECIALIZATION: Software Engineering

POSTGRADUATE: Zemin Qin

MENTOR: Associate Professor Jia Liu Lecturer Tieke He

Abstract

The judicial text is logically rigorous and clearly structured, suitable for analysis and processing using computers. Relying on the relevant policies of judicial disclosure in recent years, the available judicial corpus has grown substantially, providing data support for the application of machine learning methods in the judicial field. In order to help judicial practitioners to conduct rapid case retrieval and provide auxiliary tools for public legal consultation, this thesis studies and designs a judicial case screening system. The core function of this system is to analyze the acquired text and select similar cases from the alternative mass judgment documents in the database for users' reference, and also includes auxiliary functions such as analysis of referee documents.

The main work of the thesis is to introduce machine learning and related natural language processing techniques into the judicial field. First of all, this thesis introduces the research status of the judicial intelligence field to which the system belongs. Furthermore, the thesis introduces the word segmentation technology, text keyword extraction algorithm TF-IDF, text classification technology fastText , web application framework technology Django and other technologies related to the system construction. Last but not least,system design and implementation are completed in this thesis.

This thesis intends to classify the text through the machine learning technology fastText and further filter the similar cases based on the label results obtained by text classification. The overall design of the system is divided into four parts, including file upload module, text input module, referee paper analysis module and case screening module. The main work is as follows: The Django framework is used to implement the referee document uploading module and the text input module for realizing interaction

with the user and obtaining the text to be analyzed for the entire system. The keyword matching method and TF-IDF algorithm are used to extract the key information of the text,which helps the user to quickly understand the judgment documents. The fastText framework is used to train multiple machine learning models for classifying input texts.Further, the system filters and sorts the cases based on classification labels and text similarity. Finally, display the screening results and provide auxiliary functions such as downloading.

The judicial case screening system eventually worked well in the form of a web application. The user can access the system through a browser, perform text input, upload and analysis of the referee documents, and filter similar cases according to the input text.Through experiments, the correct rate of this system's refereeing documents analysis reached 86%, and the case screening accuracy reached 81%, which means this system has certain practicability.

Keywords: text classification, case screening, TF-IDF method, fastText framework

目　录

表　目　录

图　目　录

第一章　引言

1.1　选题的背景和意义

司法案例筛选，即以司法文本为依据，筛选相似司法案例的行为。司法案例筛选系统，包含司法案例筛选及相关的辅助功能，旨在辅助司法工作者高效处理司法案例的相关工作。

司法文本通常指裁判文书，及语言风格、表述目的与裁判文书相近的文本。这类文本用于司法过程中的事件描述，过程记录等。因而，司法文本可以分为裁判文书和非裁判文书两大类。

裁判文书是审判活动的重要凭据，由人民法院进行撰写和记载，记录了整个诉讼审理阶段的过程及结果。裁判文书不仅有清晰明确的结构要求和要素标准，同时还具有叙述逻辑完备的特点，体现着司法过程中严谨表述的要求。非裁判文书是其他个人或机构对诉讼审理过程的描述。例如，律师在和委托人交流后写下的案件过程描述、代理辩护词；法律咨询机构对客户提出问题的回复等。非裁判文书虽然不由人民法院所撰写，权威性逊于裁判文书，但一定程度上具备和裁判文书相近的要素，同样是司法过程中的记录载体。

司法文本逻辑严谨，结构清晰明确，一定程度上适合采用计算机进行辅助性的分析和处理，文本筛选即是其中常见的一种应用形式。在文本筛选的过程中，首先需要计算和分析输入文本的特征，提取其中的相关司法要素，进一步才能够依据对应特征筛选相似的文本。

因而，对司法文本进行有效处理和分析的关键，是准确地把握文本中的司法特征。在司法场景下，可考察的特征通常包括案例涉及的法律法规、案例中罚金的数额和案例量刑的尺度等等。这些特征可采用建立机器学习模型的方式进行预测或提取。

进一步地，得益于相关政策，司法公开得到不断地深化。公众得以通过网络等手段查阅不涉及国家机密及其他隐私的裁判文书。判决结果网上公开目前已成为诸多法院的基本工作。在中国裁判文书网上，已经公开了千万数量级的裁判文书数据 [1]。依托各个司法公开平台，可获取的司法文本数据量相当可观。正是依托于海量公开数据，基于机器学习方法的司法案例筛选系统才能够获取充足的模型训练素材。在司法公开的大背景下，研究司法案例筛选系统的意义主要为以下几点：

1. 为司法工作者提供判例依据

这里的司法工作者主要分为律师和法官两类群体。对于律师而言，相似案件的判决结果对辩护策略的制定是极具参考意义的，即使是经验丰富的律师在工作中也需要查阅大量相似的判例作为参考。对于法官而言，相似案例的判决结果可以视作对案件的一种“预判”。以往的相似判例能够帮助法官快速地拟定当前案件可能的判罚区间（尤其是罚金，刑期等可量化的要素）和依据的法律条文，从而提升办案的效率。

目前的司法数据库能够提供的检索服务主要停留在关键字检索的层面，仅通过关键词查找相似的案例还需要大量的人工阅读、判断和比对，将耗费司法工作者大量的时间和精力。深究其原因，主要是关键词无法将案件的全量特征做抽象，搜索到的案例相似度不足。而基于机器学习的筛选方法能够将文本整体作为输入，充分把握文本特征，帮助司法工作者获得贴切的参考判例依据。

2. 为司法研究者提供研究素材

在司法领域进行实证分析时，研究者往往需要大量的案例进行统计及分析。只有对数量充足的案例样本进行研究，才能得出具有普遍性的结论。例如，在张宝等对《侵权责任法》构成的适用争议及其消解的研究中，累计利用4328篇裁判文书 [2]进行统计及分析，最终才得以下定结论。目前，相似案例类型的研究数据仍以人工选取搜集的方式为主，需要极大的人力投入和时间投入。通过司法案例筛选系统，相关研究人员可以获得大量相似案例，这能够将研究人员从枯燥的案例检索中解放出来，将时间利用在法理分析等层次更高的工作上。

3. 为广大咨询者提供法律普及

目前社会中有大量法律咨询的需求，完全由专业的司法工作者进行解答，对于咨询者而言收费价格过于高昂。此时，以往的判例可以被提供给咨询者作为参考。司法案例筛选系统可以通过相似案例检索的方式为法律咨询者提供判例。咨询者无需对法律知识有较多的了解，通过案情简述的文本就可以寻找到案情内容相似的判例作为参考。

1.2　国内外研究现状及分析

司法案例筛选模型所属的司法智能领域在近年来有长足的发展，并获得了政策层面的支持。在国务院发布的“新一代人工智能发展规划” [3]中提出：“要促进人工智能在案例分析、法律文件阅读与分析的应用”。目前，在国内外有诸多团队从事案例筛选相关的研究和应用工作。

在理论研究部分，相关研究各有不同的切入点和侧重点。

在人工智能与司法协同理论方面，李本指出了目前人工智能在司法中应用的诸多挑战 [4],李飞系统性地论述了算法技术对司法领域的影响以及司法领域中人机协同的诠释节点 [5]。

上述研究表明，目前相关技术在司法领域中的应用主要以工作辅助为主。研究的重点应当放在辅助人类办案判决，而非替代人类审判。

在司法数据收集及预处理方面，Z Qin等从文本分析及应用的角度提出司法数据的相关考察标准 [6]。H Lian等提出司法数据的质量度量方法 [7]及利用社交媒体数据和司法数据构建知识图谱的方法 [8]。G Lame等提出使用NLP技术识别法律本体组件的方法 [9]。MR Talib等提出基于子句元素对司法系统语料库进行挖掘的方法 [10]。

根据上述研究可知，目前对司法文本进行处理的方法主要有两种：基于one-hot方法的文书向量表示和基于word-embedding文书矩阵表示。两种模型的应用场景不同，当数据量较小并且数据质量较高时，基于one-hot的模型效果更好，能够有效体现对案件要素的注重；当数据量较大足以支撑训练模型时，基于word-embedding的表示方法因为其保留对上下文语义的理解而效果更优。在实际使用中，需要建立合理的数据收集机制，并依托数据的质量和数量，选择合适的文本表示方法和挖掘方法。

在具体司法场景中机器学习（包括诸多人工智能技术）方法的应用方面，T He等提出采用主题模型对司法案例罚金预测的方法 [11]。Z Liu等对机器学习模型在司法领域的预测性能进行了比较 [12]。E Ash提出采用多种机器学习方法（包括深度学习手段）预测司法文本对应判决尺度的方法 [13]。

根据上述研究可知，目前机器学习技术在司法领域的应用主要以分类（文本标签预测）的形式为主。在有合适训练数据的情况下，各类机器学习手段在具体细分的分类或预测任务上均能取得良好的效果。

在应用研究部分，有诸多创业型企业专注于司法智能领域。

在国外，美国企业IBM于2016年推出了一款基于Watson的法律咨询应用ROSS [14]。这一应用基于语音识别，能够从用户的提问中提取关键信息进行检索，获得相应的解答，并且提供ROSS依据的原始案例文本。ROSS并不是一个法律领域里通用的助手，目前主要专注于破产处理领域。ROSS实质上是提供语音交互接口的关键字类案检索工具。

在国内，幂律、华宇元典等团队同样开展了司法智能的应用研究。例如，元典团队推出产品“元典智库”，提供基于关键词的案例细分检索工具；推出产品“法官怎么判”，提供基于大数据分析的法官判决偏好统计。上述工具以传统检索和统计方法为主，相对成熟，目前已经有初步的商业化应用。

各团队的工作各有特点和侧重点。总的来看，目前与案例筛选相关的司法智能工具及系统，主要有以下2种实现形式 [15]：

1. 限定检索范围的关键字检索

此类方法是对传统的检索系统的改进。传统的检索系统通常包含关键字、标题、全文等范围的检索。在目前流行的检索系统中，拓展了可选定的检索范围。这类系统对检索库中的文本做结构化抽取。除了传统的案件标题外，还包括了案由，诉辩陈词（原告诉称、被告辩称），法院调查及观点（本院查明、本院认为），判决结果（裁判结果、引用法条），涉案主体（法官、当事人、律师、律师事务所、案号、法院）等要素。检索库中的文书依据上述的要素被结构化抽取。在检索时可以选择其中的任意类型进行检索，有效的定位需要检索的范围。由于司法文本中存在标准的格式和关键字，例如“本院查明……”。在实际抽取时，通常只需要截取关键字所在的段落就能够完成抽取。

此类方法成熟度较高，但未能与目前新兴的自然语言处理技术产生有效结合，仍停留在关键字匹配阶段，能够处理的文本长度较短，灵活性较差，仅适合部分简单场景。

2. 标签预测及基于标签的推荐

此类方法需要人工标注检索库中的司法文本作为训练素材，通常需要标注的标签包括诉讼类型、案由、地点、审理法院等。完善标注后，训练对应的标签预测模型。在推荐时，先通过模型获得待推荐文本的标签，再依据标签推荐同标签或标签相近的文本。

此类方法目前并无成熟的产品问世，尚处于摸索阶段。各类机器学习方法在司法领域中的实际应用效果亦有待观察。

本文主要探讨司法案例筛选系统。目前期待的核心效果是：输入一段案例的案情描述文字或裁判文书文本，系统根据输入文本的特征推荐相似的案例。推荐的案例是已经完成判决的相似案例，能够对当前的案件判决提供参考。对于这样形式的系统，目前并没有过多可供参考的应用范例。

1.3　课题来源及主要研究内容

本文课题源自于某国家重点研发计划，是这一课题重要的研究内容之一。目前，以大数据及人工智能为代表的新兴技术已经在各个领域中得到初步应用。司法领域是一个非常注重文本的领域，任何诉讼、纠纷的解决，都保留行文规范严谨的裁判文书作为书面记录。因而，自然语言处理技术在司法领域具有充足的语料可供研究使用，具备开展研究的条件。

司法案例筛选（也称做类案推荐），就是其中一个值得研究的问题。在实际的案件诉讼中，法官和律师都需要根据当前的案情查阅相关的以往判例。同时，类案类判实际上也是司法公正和公平的体现。最高人民法院公开发布的指导案例，一定程度上就是通过指导性案例的方式推广合理统一的判决尺度。因而，类案的检索是司法实务工作中的一个常见且重要的环节。

目前，类案的检索一方面依靠司法工作者自身的经验，根据对案件的判断从自己处理或者研究过的案例中检索；另一方面，通过关键字检索的方式在案例文本库中检索。总体而言，目前在类案推荐方面，司法工作者需要花费很多时间。工作中配套的软件工具，并没有获得当前新技术的支持。

本文主要研究的课题是将机器学习的方法用于实现案例分析和案例筛选，以帮助司法工作者减轻工作量。具体而言，本文将研究一套具有一定实用价值的司法案例筛选的方法，并完成其整体系统设计及实现。主要的研究点在于摆脱传统关键字检索的查找方式，以文本整体作为系统输入进行案例筛选。本文拟采用的机器学习方法，通过大量标记过的案例数据对文本分类模型进行训练，并依据分类的结果筛选出相似的案例。这一方法具有语义敏感度高的特点，即使文本中没有检索库中的关键字，也能够给出语义层面相似的筛选结果。

1.4　本文的工作

本文的研究在司法信息化的背景下展开。目前业界对自然语言处理的研究已经取得了长足的进步，并且开源了非常多的工具和框架。但是具体到司法领域中，对新兴的自然语言处理技术应用得并不多。在实际应用中，司法工作者也需要新的技术来帮助他们减轻工作量。

基于上述考量，本文的工作重点在于找到自然语言处理技术在司法实务中应用的具体方式方法。具体的着眼点即为解决案例分析和案例筛选（类案筛选）的问题，并通过相关司法的数据集进行模型训练，实现可运行的原型系统。

本文具体工作如下：首先，介绍系统所涉及的各类文本处理技术，并比较目前主流的机器学习方法在司法领域应用的优劣势。进一步，结合司法实务需求进行系统整体设计，将司法案例筛选化归为可用相关机器学习模型解决的子问题。采用机器学习方法，通过构建文本分类器完成对案例特征标签的预测，并设计相关筛选算法依据标签完成案例筛选的功能。同时融合案例分析等辅助功能，增强系统的可用性。完成系统设计后，将展开论述整个系统的具体实现的过程。最后，对完成的原型系统进行测试调整，确保这一司法案例筛选系统能够稳定且高效的运行。

1.5　本文的组织结构

得益于算力的不断增长和海量数据的公开，基于机器学习的自然语言处理技术正方兴未艾。本文将采用新兴的机器学习技术解决司法领域中案例筛选的问题，并以此衍生为可用的原型系统。本文组织结构如下：

第一章引言，介绍了项目背景及意义、国内外在该方向的研究现状、本文所属课题来源、本文的主要研究内容及主要工作。

第二章技术综述，将叙述本文所涉及的技术和开源框架，包括分词技术、TF-IDF算法、文本相似度计算、FastText框架和Django框架等。

第三章系统需求分析与概要设计，将提出项目基本需求，并对项目总体设计思路进行了概述，对项目模块进行了划分、从多个层面对系统进行设计。

第四章系统详细设计及实现，将在需求分析的基础上，重点阐述了系统各个模块的具体实现方法和过程。

第五章总结与展望，将总结论文期间所学习的理论和所做的工作，为系统未来的功能扩展和推广使用作进一步展望。

第二章　技术综述

2.1　文本处理技术

司法文本处理，主要是针对中文文本的自然语言处理，有诸多成熟技术可供参考和使用。目前，文本处理技术的基础大多源自于英文文本的处理。中文和英文文本处理虽然有诸多共通的内容，但是依旧有各自独特的文本处理环节[16]。例如，英文文本中需要进行大小写的转换，需要去除一些表述中的符号，这些都是中文文本处理中不需要处理的内容，但中文文本处理中也有需要额外处理的环节。

与英文单词之间有空格不同，中文的词之间并没有分隔。因此，处理中文文本需要在提取过各种非文本符号后，进一步将句子切分为词语 [17]。中文中的词语往往才是表达意思的基本单位，实现了分词功能后，才能够更为方便地提取出文本的关键词，进行词向量转化并进行后续诸多处理。接下来将介绍一些中文分词、关键词提取以及文本相似度计算等方面的基本技术。

2.1.1　中文分词算法

分词，可以与短句切分的过程相类比。具体是指在一个无切分的短句中根据意群将词语切分出来。目前在自然语言处理领域，中文分词主要有以下三类算法 [18]：

（1）基于匹配的分词方法

基于匹配的分词方法是一种规则相对简单，但是形式机械的分词方法。这种分词方法按照一定的规则将待分词的文本与内置机器词典中的词条进行对比。若在词典中找到待分词的文本中的内容，则认为匹配成功，切分出对应的词语。这种方式可以选择不同的匹配方向。一般而言，从左到右的方向（与当前的阅读方向相同）称之为正向最大匹配法，而从右到左的方向（与当前的阅读方向相反）称之为逆向最大匹配法。

切分方法除了由切分方向区分外，还可从切分次数加以分别。最少切分法对应每句文本中切出的次数最少的方法，是一种在多种切分都可行时选择最终切分的辅助方法。为获得更好的准确率，还可以将以上方法同时使用。例如，同时使用正向最大匹配法和逆向最大匹配法，这种方法也被称为双向匹配法。在实际应用中，基于匹配的分词方法准确度尚未达到理想程度。实际使

用匹配分词时，往往还需要利用文本中其他的信息来作辅助切分。例如，优先在待分词的文本中找出一些较为明显的“断点”词汇，例如句末感叹和语气词“啊”,“呀”，句中虚词“的”等。利用这些词对文本先进行一次切分，在虚词切分的基础上再采用匹配分词的算法，可以显著提升分词的成功率。

在这一方法下，还衍生出词性作为分词的依据的辅助方法，在分词时参考词性信息，同时对切分出的词进行词性标注。采用这样融合的方法也能够显著提升分词的准确率。

（2）基于理解的分词方法

基于理解的分词方法是将语义理解加入到了分词中，通过句法语义先对句子进行分析，使计算机达成对句子的理解，进而完成分词。这种方式尝试完全用计算机模拟人的思维以及理解模式，以这种方法进行分词时需要调用大量的语义信息和句法信息。在中文文本总体较为复杂且语义环境相对模糊的情况下，难以获得很好的实施。相关的工具和开源框架大多已经终止，且进一步研究的可能性也较为渺茫。

（3）基于统计的分词方法

基于统计的分词方法是目前的效果较好的分词方法。一般来说，各个字符在文本中相邻出现的次数越多，构成一个词的概率就越大。因此字之间彼此相邻出现的概率（或者频率）能够较好反应构成词的可信程度。在基于统计的分词方法进一步细分，采用隐马尔可夫链（Hidden Markov Model, HMM）统计模型的分词方法是目前的最主流的方法 [19]。马尔可夫链描述了一种状态序列，是满足马尔可夫性质的随机过程。

在中文分词模型中，输入文本和输出的分词结果都被当作一个序列。在这种情况下，分词问题就被化归为序列标注问题。输入的文本为可观测的序列，分词结果为不可观测的状态序列。分词结果可能有四种情况，如下表 2.1所示：

表 2.1: 分词结果状态表

类型	表示符号	前缀	后缀
词首	B（begin）	S E	M E
词中	M（middle）	B M	M E
词末	E（end）	B M	B S
单字成词	S（single）	S E	S B

由表 2.1可知，要完成对序列的分词，就是将文本当做观测，获得各个文字对应的状态序列，即可获得分词结果。

在马尔可夫链的假设中，分词状态序列中的每个状态值只取决于该状态前的有限个状态。而隐马尔可夫链则是马尔可夫链的一种特例和推广 [20]。在隐马尔可夫链中，状态序列是不能被直接观测到的，故称之为隐（Hidden）。尽管状态序列不可被直接观测，但仍有可直接观测的序列，称之为观测序列，这些可观测的向量通过一定的概率密度分布展现为各种状态。

在模型中，除了涉及可观测的状态序列和隐藏的状态序列，还应包含了各个隐藏状态的初始概率，隐藏状态的转移概率，以及从可观测的序列到隐藏状态的转移概率（又称之为发射概率） [21]。这些文字对应的初始概率，转移概率和发射概率是隐马尔可夫链需要训练的参数。这类参数可由大量语料统计后获得，针对不同语料类别可训练不同参数。

基于上述的理念与方法，有诸多如jieba，THULAC等开源的分词工具可供选用。本文将采用基于HMM方法的分词工具jieba对文本进行分词处理。

2.1.2　TF-IDF关键词提取

在本文所研究的案例筛选系统中，包含对案例进行分析的环节。提取文档关键词是其中的一项主要功能。提取文档的关键词有多种方法，所依赖的具体算法原理各不相同。在文档中直接提取出现过的高频词是最为直观的思路。在中文文本中，纯粹的高频词一般都是“的”、“啊”等虚词。提取高频词作为关键词，同时还要求被提取词语为高频实意词。TF-IDF算法就是这样一种简单直观的高频实意词提取算法 [22]。

TF（text frequency）指词频，即某一词出现在文本中的次数。由于不同文章长短不同，仅统计词频会有失偏颇（长文本的词频一般而言数量更多）。因此一般会对词频进行归一化。通常采用某一词在文本中出现的次数与文本的总词数的比值来确定，也有部分学者及相关文献的记载中采用某一词在文本中出现的次数与文本中所有词语出现次数极值的比值作为TF值。IDF（inverse document frequency）指逆文档频率。用来描述一个词的常见程度，需要对大量文档进行统计从而获得每个词的IDF值 [23]。

一般来说，常见词在各类文章中均有出现，意味着包含该词的文档数量较多，逆文档频率（IDF）应当较小。IDF值较小说明该词的稀有度很低，对于文档的代表性也很差。反之，若某个词在整个语料库出现的次数很少，则说明该词的对于自身文本而言代表性很强，适合作为所在文章的代表和标签，对应的IDF值应当较大。

依据TF-IDF值，对文档中每个词的TF-IDF值进行降序排列，即可获得各个词的关键性排序，关键词提取通常选取排序靠前的若干词。TF-IDF算法的总体

思想虽然简单，但实际的使用效果依旧能维持较高水准。具体实施和部署的过程也并不繁琐，没有主题模型等方法那么复杂的训练过程。但是，往往词频并不是关键词的唯一标准。尤其在语义层面，出现次数多的词并不代表这个词的重要性高。即使有IDF值对词的代表性进行修正，但距离真正语义层面上的关键词仍然还有一定的偏差。

为了克服上述缺陷，对于TF-IDF算法，需要构建有针对性的语料库（本文中将使用司法领域的裁判文书）。针对新加入的文档，需要及时的修正IDF值，使得语料库的针对性越来越强 [24]。针对有特殊意义的词，可以适当地调高其IDF值。对于某些重要性极高的词，简单地调高IDF值的意义并不大，可以将这类词收集进入词表。在采用TF-IDF算法提取前，以词表中的词为目标，在文档中进行精确匹配，一旦匹配成功即优先将其作为关键词，通过这样手法进一步完善关键词提取的效果。

2.1.3　基于TF-IDF的文本相似度计算

系统通过文本分类模型获得文本标签后，依据标签在案例数据库中检索筛选同标签的案例。但对于同标签的案例，仅依据标签信息无法对其进行排序。此时，需要引入文本相似的计算解决这一问题 [25]。

通过上述TF-IDF算法的统计，可将文本转化为向量。因而，文本相似度计算可进一步转化为向量相似度的计算。向量相似度的计算常用的方法包括包括余弦相似性、欧式距离等：

1. 余弦相似性 [26]

余弦相似性，也称余弦距离。计算两个文本向量夹角的余弦值作为两个文本向量差异的大小的度量。设两向量为a，b，则其余弦距离如下式所示：

$$sim(a,b) = \frac{< a,b >}{|a|\,|b|}$$

由于文本向量均为正，因此余弦值0到1之间。0度角的余弦值为1，且余弦函数在0-90度范围内单调递减，因而余弦值越高代表文本相似度越高。

2. 欧氏距离 [27]

欧氏距离是欧氏空间中两点间的距离公式。设两向量为a，b，则其欧氏距离如下式所示：

$$sim(a,b) = \sqrt{(a-b)(a-b)^T}$$

欧氏距离为0，代表文本向量完全重合。欧氏距离值越大，代表文本相似度越低。度量相似度时，有时也采用1/(1+欧氏距离)的归一化手法。

余弦相似性相比欧氏距离，有归一化的特点，且其值与文本相似度呈正相关。因而在本文中，将采用余弦相似性作为文本相似度的度量。

2.2　文本分类模型

在对司法案例文本进行处理和分析后，需要对其进行文本分类，并依据分类的结果进行案例筛选。目前可用的文本分类的模型较多，包括采用支持向量机方法的LibSVM [28]、采用CNN模型的TextCNN以及采用浅层神经网络模型的fastText等。目前以TextCNN为代表的深度学习模型和专用于文本数据的fastText效果最为出色。本文对上述两种模型进行了实验尝试，简述如下。

2.2.1　TextCNN模型

TextCNN将在图像领域中取得非常优异效果的卷积神经网络应用到了文本层面 [29]。通过“卷积”以及“最大池化”两种在图像识别中效果上佳的手段对文本进行处理。由于卷积神经网络卷积核的存在，因而需要将文本转换为矩阵的形式予以处理。由于文本本身可以看做由一系列词组成的向量，因此将词扩充为向量后，文本即可视为矩阵。模型对矩阵化的文本进行卷积和最大池化（max pooling）后，再通过全连接层的Softmax进行结果输出。

TextCNN能够有效的对文本的特征词进行感知，从而进行对文本的分类。然而，最大池化（max pooling）[30]在对文本关键词进行高效集约的同时，也丢失了文本的结构信息。对于同样一篇文本而言，打乱文本的顺序对TextCNN的结果输出影响甚微。TextCNN模型只能感知到文本中影响分类的关键词是否存在，而无法感知到关键词出现的次数和顺序。因而，对于裁判文书中的先后因果等要素，在TextCNN的模型处理中较难妥善处理。

同时，由于卷积结构相对复杂，TextCNN在模型训练方面的开销相对较大，训练模型的时间很长。并且调用TextCNN模型事先需要加载的语料库和词袋都非常庞大，使得单次调用TextCNN模型的时间较长（实际实验中超过10s），超出了系统要求的响应时间。TextCNN深度学习模型效果虽然优异，但实时性无法达到期待的标准。因而，即使TextCNN在合理的训练和选择下能够有非常出色的表现，但在本文所述的系统中不采用TextCNN作为分类的模型。

2.2.2　fastText模型

fastText是目前另一款适用于文本分类的开源框架，源于Facebook AI research中开源的项目。fastText的结构相比TextCNN更为简单，因而在训练和调用的速度上与TextCNN相比很有优势，并且分类的精度并未逊色。fastText采用单层神经网络的结构，文本结构信息的丢失同样很严重。但通过引入N-gram向量后，词与词之间的关系得以保留，使得文本结构信息一定程度上得以保留，弥补了单层神经网络的缺陷。

在这一情况下，本文选择fastText作为文本分类的框架。这一框架有两项主要特点：训练快速和文本专用。两项特点在框架的名称fastText中有直观的体现。fast代表这一框架在模型训练和调用预测上相对于其他的文本分类模型速度更快。即使是大型数据，也能够在一台硬件配置达到当前主流水平的机器上以分钟级的时间代价训练完成 [31]。训练出模型的效果与深度学习分类器相比精度差别不是很大，甚至优于部分的深度学习分类器，如图 2.1所示 [31]：

	Yahoo		Amazon full		Amazon polarity	
	Accuracy	Time	Accuracy	Time	Accuracy	Time
char-CNN	71.2	1 day	59.5	5 days	94.5	5 days
VDCNN	73.4	2h	63	7h	95.7	7h
fastText	72.3	5s	60.2	9s	94.6	10s

图 2.1: fastText与深度学习对比图

fastText另一特点，在于“text”。这是一个完全用于文本分类的框架。可用于句子（短文本）分类和长文本分类等领域 [32]，在文本倾向性和文本分类标签预测等典型问题上有着非常优异的表现。并且这一框架能够通过学习词语的向量表示来处理多种语言的文本,因而对于中文的司法文本同样适用。

fastText模型主要由三部分组成 [33]：浅层神经网络模型架构、层次Softmax和N-gram特征。在模型架构方面，fastText采用了与word2vec训练词向量所用的CBOW模型相似的浅层神经网络模型（如图 2.2所示）。

与深度神经网络隐藏层包含多层不同，fastText所采用的模型隐藏层仅有一层，即整个模型是一个三层架构：输入层、隐藏层、输出层。这一模型架构与word2vec训练词向量的CBOW模型的区别在于输出层，在fastText模型中，输出层是预测的文本标签，而在CBOW模型中预测的是中间词 [34]。

为了减少在类别和语料增加时的计算开销，fastText模型采用了基于霍夫曼编码树的层次Softmax。在输出预测类别不均匀的情况下，通过霍夫曼树构建的树形结构能够使得频繁出现的类别所属的树型结构比其他类别的深度更小。这

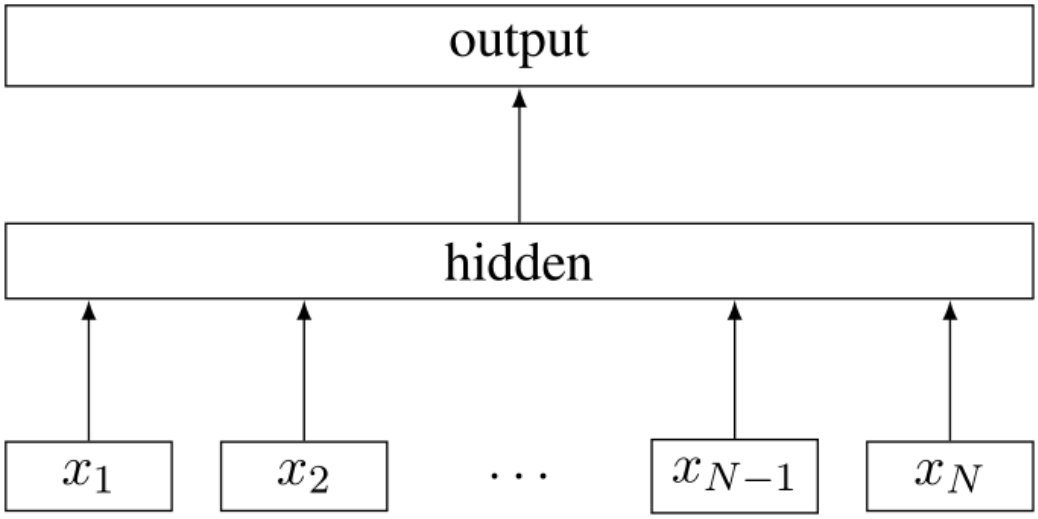

图 2.2: fastText网络结构图

也是fastText计算的效率较高的原因之一。

同时，fastText还保留了词袋特征，即N-gram特征。这一特征实质上是对语序关系的一定保留和刻画，通过N-gram特征保留的部分原文语序信息 [35]，使得fastText拥有了更为出色的预测结果。

fastText作为一个专门用于文本分类的框架，有着丰富可调节的参数，能够满足本文所研究的案例筛选模型的文本分类需求。在实际的系统中，fastText框架能够在相对较短的时间内将对预处理后的文本进行分类，为后续的案例筛选提供依据。

2.3　Django框架

为了使案例筛选系统具有较好的拓展性和可用性，本文决定采用Web应用程序的方式实现。Web应用程序，即为通过网页访问的应用程序。用户通过浏览器即可访问，不需要额外安装任何环境。

2.3.1　Django简介

Django是一个在前端方便易用，能够让使用者快速生成功能强大，安全可靠代码的web框架 [36]。Django使用Python构建并且隐藏封装了大量的功能。这将帮助程序开发者快手上手，使用时不必花费过多精力去关注web编程的具体实现，从而注重于具体程序和功能的开发。Django具备完善且活跃的用户社区，

并且开源了诸多代码库用于在Django项目拓展使用。基于Django框架的上述特性，本文将采用Django作为web应用的生成框架。

2.3.2 Django模板

Django支持网页模板加载、级联样式表渲染等web功能。级联样式表（CSS）是用于描述html文档的表示的样式表语言 [37]。CSS同时也是大多数网站用于创建具有视觉吸引力的网页使用的基础技术，同时也经常用于Web应用程序的用户界面以及许多移动应用程序的用户界面。CSS的主要功能是使文档内容与文档呈现分离，可以设定包括布局，颜色和字体等要素。这种设计上的分离可以提高内容的可访问性。同时，采用CSS样式表可以使得多个网页共享同一个CSS样式，对于一处样式的改变可以全部应用到相关的网页上。

在Django模板中还提供了多种标签实现与后端的耦合。“{{}}”标签用于变量的传递。例如，在模板中出现：“{{var}}”，在加载时，模板会将名为‘var’的变量插入“{{var}}”的位置。“{%%}”用于标记逻辑块，用于在页面中实现各类逻辑语句。例如，通过{%for%}标签等可以在模板中设置循环语句等等。

Django模板还支持开发者自定义标签，使得模板功能更为完备和强大。

2.3.3 Django工作流程

由Django生成web应用的工作流程如下：

首先，由使用者请求web应用页面的url。这一请求会到达Django的Request Middleware (Django的请求中间件)。中间件的作用主要是发起response请求，有时会附加预处理的操作。进一步，由URLConf通过Django项目下的url.py文件，结合请求的url寻找对应的视图。此时视图中间件被激活，同时视图中的函数被调用(view.py)。经过视图中函数的处理，Request得到响应，对应的HTTPResponse被发送到Reponse中间件。最后，Response可依据设置直接将值返回浏览器，也可返回Template渲染的页面。经过这一流程后，Django web应用完成了一次用户响应。web应用的功能也需要整合到这一流程之中。

2.4 本章小结

这一章主要介绍了开发案例筛选系统所用到的相关技术及对应的开源框架。首先介绍了文本处理技术，包括中文分词，TF-IDF关键词提取和文本相似度计算技术。接着比较了用于提供案例筛选依据信息的TextCNN模型及浅层神经网络模型框架fastText。最后介绍了用于构建web应用的框架Django。

第三章　系统需求分析与概要设计

3.1　系统总体规划

本系统由文件上传模块、文本输入模块、裁判文书分析模块和案例筛选模块四部分模块组成。其中，案例筛选模块是系统的核心部分，在这一模块中将实现司法案例筛选的主要功能；裁判文书分析模块对上传裁判文书进行分析，帮助用户快速理解裁判文书；文件上传模块和文本输入模块用于和用户的交互，为整个系统的获取需要分析的文本，提供基础的支撑服务。

3.2　司法案例筛选系统需求分析

3.2.1　司法案例筛选系统功能需求

在实际司法实务中，司法工作者需要对大量的司法文本进行处理，包括文本研读分析、相似案例查找等工作。与此同时，也有诸多民众有着大量的法律咨询的需求。在目前律师咨询费高昂的情况下，大多数人选择是自行查找已有的类案或判例，通过以往的案例经验来参考当前自身面对的问题，以确定是否采取进一步的行动（放弃诉诸法院，达成和解；或意识到问题复杂性，决定选择聘请专业律师等等）。

司法案例筛选系统，需要以在线web应用的形式，为广大司法工作者和有咨询需求的民众远程地提供案例分析，案例筛选的服务。这一系统为了确保受众能够较为方便的获取，应以在线形式呈现，即点即用，无需安装任何的依赖环境。用户访问这一应用后，可以通过文件上传的方式向系统传递想要分析的裁判文书，并获得依据文书筛选出的相似案例；也可以通过文本框输入的方式输入简短的案情描述，获得对应的案例筛选。

裁判文书是在人民法院审理过程中用于记录审理过程和结果的书面载体[38]。人民法院通过裁判文书对涉案当事人实体进行权利与义务的分配和确定。因此，裁判文书通常拥有完整的结构，各项要素缺一不可，同时还需要保障行文中不能存在前后矛盾等不严谨的瑕疵，适合做结构化的程序分析。

近年来，随着司法公开的深入，有大量的生效的裁判文书在隐去个人隐私信息后上传到中国裁判文书网 [39]，这无疑进一步提升了裁判文书的关注度与普及率。由裁判文书衍生出的分析和相似案例筛选的业务也受到了更为广泛的关注。综上所述，系统的主题功能需求可归纳下属几点：

裁判文书分析功能，需要帮助系统使用者快速地理解裁判文书。对于上传裁判文书的分析，需要解析裁判文书的各项关键信息，包括裁判文书的文书类型、判决法院、案件的案由（或审判的罪行）以及裁判文书中本身的关键词。

案例筛选功能，需要通过用户输入或上传的文本，筛选出相似的案例文本。相似需要体现在案例文本所描述的事件相似（案由、罪行等相同或相近）并且情节严重性（刑期）相似。通过两个维度上的相似，确保筛选案例的相似。

司法案例筛选系统需要集成文件上传和文本框输入两种交互模式，并且提供裁判文书分析和案例筛选功能，以web应用的形式在线提供服务。用户完成案例筛选后还可对浏览过的案例进行下载。

系统主要的功能性需求整理如表 3.1所示：

表 3.1: 系统功能性需求列表

需求ID	需求名称	需求描述
R1	远程文本框输入	用户能够使用远端浏览器访问系统的输入页面 用户点击提交后，后台服务库获得用户输入的文本 文本框需支持文字的复制粘贴
R2	远程文件上传	用户能够使用远端浏览器访问系统的上传页面 用户点击“选择上传文件”按钮打开文件管理器 点击“上传”按钮，将文件上传至服务器端 上传文件支持的格式为txt
R3	裁判文书分析	服务器获取上传的文件，进行分析 分析的内容包括文书类型、判决法院、涉及法律 案件的案由（或审判的罪行）以及关键词
R4	获取筛选案例	服务器获取待处理的文本 调用相关的分类模型对文本进行标签预测 依据标签及文本相似度筛选出相似案例 展示原案例及相似案例的摘要列表
R5	详情查看	获得筛选结果列表后可选择查看备选结果全文 依据选择的文本继续进行相似案例筛选
R6	文件下载	用户在浏览案例时，可以下载当前案例文本 可以批量下载相似案例列表的文本
R7	裁判文书存储	服务器端储存没有分析过的案例及对应的分析结果 对于已分析过的案例直接展示储存的分析结果

3.2.2　司法案例筛选系统非功能需求

表 3.2: 系统非功能需求列表

时间特性	用户任何页面跳转操作响应时间不超过0.5second
	用户使用案例分析或案例筛选操作响应时间不超过5second
负载	支持1位用户远程同时访问

3.2.3　司法案例筛选系统用例设计

根据系统功能性需求，系统的用例图如图 3.1所示：

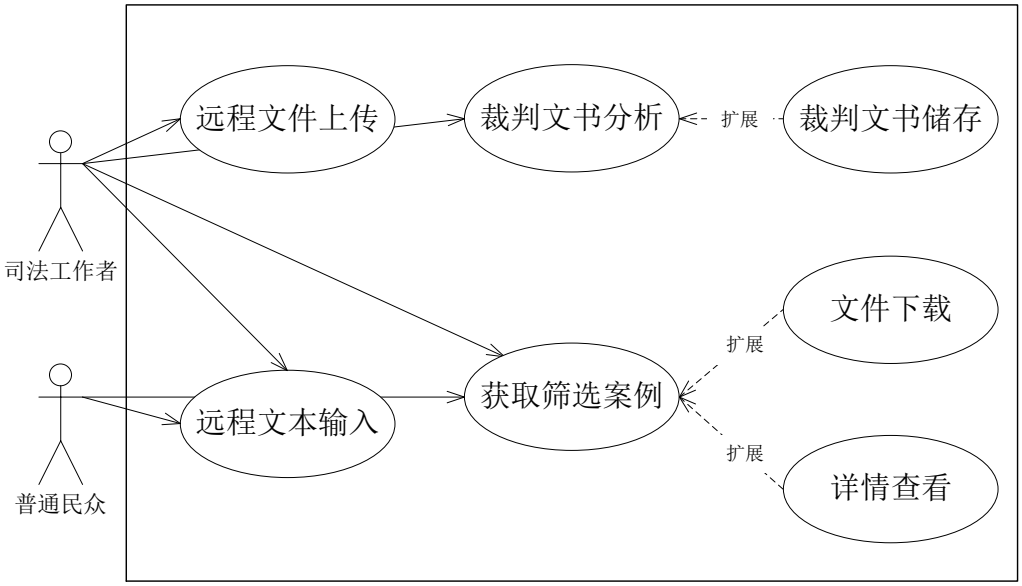

图 3.1: 系统用例图

用例图中共涉及7个用例，分别是远程文本输入、远程文件上传、裁判文书分析、获取筛选案例、详情查看、文件下载和裁判文书储存。

其中，远程文本输入用例和远程文件上传用例定义了文件上传和文本框输入两种交互功能。裁判文书分析用例定义了裁判文书分析功能：需要帮助系统使用者快速地理解裁判文书。裁判文书储存用例定义了对裁判文书及其分析结果储存的功能。获取筛选案例用例定义了案例筛选功能：需要通过用户输入或上传的文本，筛选出相似的案例文本。详情查看用例和文件下载用例分别定义了对筛选案例的原文本详情查看和下载的功能。7个用例的具体用例描述如下：

1. 远程文本输入

远程文本输入是系统与用户交互的两个基本用例之一，描述的是用户通过浏览器页面输入案情描述。具体用例如下表 3.3所示：

表 3.3: 远程文本输入用例状态描述表

ID	UC1
名称	远程文本输入
用例属性	创建者：admin
参与者	普通用户
目的	向文本框输入欲进行案例筛选的案情描述
描述	用户通过浏览器页面上的文本框填写案情描述
优先级	高
触发条件	用户进入填写页面
前置条件	用户成功访问web应用的对应的url
后置条件	服务器获取案情描述，进入案例筛选结果展示页面
正常流程	1.用户访问web应用中远程文本输入对应的url
	2.用户在文本框中填写案情描述
	3.用户点击“提交”按钮，文本框内的文本上传 服务器进行处理并返回筛选结果页面
异常流程	3. 用户输入的文本长度超过一次POST操作的极限大小 系统弹出错误页面，用户需返回重新输入合法长度

2. 远程文件上传

远程文件上传也是系统与用户交互的两个基本用例之一，主要由司法工作者处理较长的文书时使用。远程文件上传需调用文件管理器进行文件选择，并且存在文件格式错误的问题，具体用例如表 3.4所示。

3. 裁判文书分析

裁判文书分析是系统的主要功能，在用户上传合法的裁判文书后自动进行，对裁判文书的关键信息进行解析。此功能主要提供给司法工作者使用。用例描述的是用户上传了合法案例后对案例关键信息的查看。具体用例如表 3.5所示。

4. 获取筛选案例

获取筛选案例是系统的核心功能，在用户上传合法的裁判文书或用户在文本框输入内容后自动进行。描述的是用户上传或填写了合法文本后，系统筛选相似案例，提取其摘要信息并形成列表返回。具体用例如表 3.6所示。

表 3.4: 远程文件上传用例状态描述表

ID	UC2
名称	远程文件上传
用例属性	创建者：admin
参与者	普通用户
目的	向服务器上传欲进行案例分析和筛选的裁判文书
描述	用户通过浏览器页面上传待分析的裁判文书
优先级	高
触发条件	用户进入上传页面
前置条件	用户成功访问web应用的对应的url
后置条件	服务器获取上传文件，进入分析结果展示页面
正常流程	1.用户访问web应用中远程文件上传对应的url
	2.用户点击“选择上传文件”按钮，打开文件管理器
	3.用户在文件管理器中选择需要上传的文件（txt格式）
	4.文件上传, 服务器进行处理并返回展示页面
异常流程	4.用户选择的文件类型错误或选择的文件过大 系统弹出错误页面,用户需返回重新合规的文件

5. 详情查看

相似案例的筛选具有强烈的情境性和主观性。在具体的司法情境下，相关的司法从业者对相似案例的认知和理解是不同的，因而需要不断迭代和选择才能筛选出在当前情境下有参考意义的案例。在将筛选出的相似案例列表展示给用户后，需要根据用户的反馈作进一步的推荐。因而，用户选择查看案例详情后，系统将以备选案例的文本作为案例筛选的输入继续进行相似案例的筛选，并展示作为进一步推荐。用例与获取筛选案例相似，故不另设用例。

6. 文件下载

用户筛选到相似案例后通常需要将案例进行保存，文件下载功能对于用户而言是非常实用的功能。同时，对于筛选出的相似案例列表。还应提供批量下载的方式，用例如表 3.7所示。

7. 裁判文书储存

裁判文书储存是用于储存新收录分析的裁判文书。当系统计算并检索当前上传裁判文书的MD5码后，若系统未收录则自动存储，无需用户操作，因此不另外设置用例描述表。

表 3.5: 裁判文书分析用例状态描述表

ID	UC3
名称	裁判文书分析
用例属性	创建者：admin
参与者	普通用户
目的	用户查看裁判文书的分析结果
描述	用户通过浏览器页面查看裁判文书分析结果
优先级	高
触发条件	用户点击文件上传页面的“上传”按钮
前置条件	用户已选择格式正确且大小合法的上传文件
后置条件	用户可以查看案例筛选结果展示界面，或返回继续上传
正常流程	1. 用户点击“提交”按钮，文本框内的文件上传
	2. 服务器进行处理并返回裁判文书分析结果页面
异常流程	1. 上传的裁判文书不是可解析的标准格式 系统返回错误信息

表 3.6: 获取筛选案例用例状态描述表

ID	UC4
名称	获取筛选案例
用例属性	创建者：admin
参与者	普通用户
目的	用户获取筛选案例的分析结果
描述	用户通过浏览器页面查看筛选案例的展示
优先级	高
触发条件	用户上传或输入合法文件或文本
前置条件	无
后置条件	用户可以查看筛选案例的详情，或进行下载
正常流程	1. 用户点击“提交”按钮，提交对应的文本或文件
	2. 服务器处理并返回筛选案例摘要列表，同时显示原文
异常流程	1. 上传的内容不是可解析的格式 系统返回错误信息

表 3.7: 文件下载用例状态描述表

ID	UC5
名称	文件下载
用例属性	创建者：admin
参与者	普通用户
目的	用户下载查看案例的文本
描述	用户通过下载或批量下载案例的文本
优先级	高
触发条件	用户点击对应下载链接
前置条件	用户进行案例筛选
后置条件	用户可以继续查看筛选案例的详情下载
正常流程	1. 用户点击“下载”按钮，浏览器开始下载对应的文件
	2. 用户点击“批量下载”按钮 系统将案例文本写入同一文件提供下载
异常流程	1. 获取下载文本内容失败或生产批量下载链接失败 系统返回错误信息

3.3 司法案例筛选系统设计

3.3.1 系统逻辑设计

根据上文分析的需求和设计的用例，本节依照系统运行逻辑，将系统进一步划分模块，主要分为文件上传模块、文本输入模块、裁判文书分析模块和案例筛选模块四部分。同时，裁判文书分析模块和案例筛选模块因为功能相对复杂，又进一步划分出了对应的子模块，如图 3.2所示。

文件上传模块和文本输入模块是两个相对比较简单的模块。这两个模块负责用户和系统的交互工作。文件上传和文本输入是两种互相补充的交互手段。当用户需要检索的内容相对简短时，可以采用文本输入的方式交互，比在本地编辑后上传的方式更为快捷。同时，需要对检索的内容进行修改时，通过文本框可以直接进行编辑。这样的功能适合用户连续迭代检索内容，直至找到需要的结果。文本输入模块中点击提交后，会调用案例筛选模块对输入的文本进行分析（输入的文本结构一般不符合裁判文书的标准，因而不调用裁判文书分析模块）。

而文件上传模块主要适用于检索内容较长，需要通过文档进行编辑和保存的情形。这一功能多使用于用户想对手头的裁判文书进行相似案例的检索。由

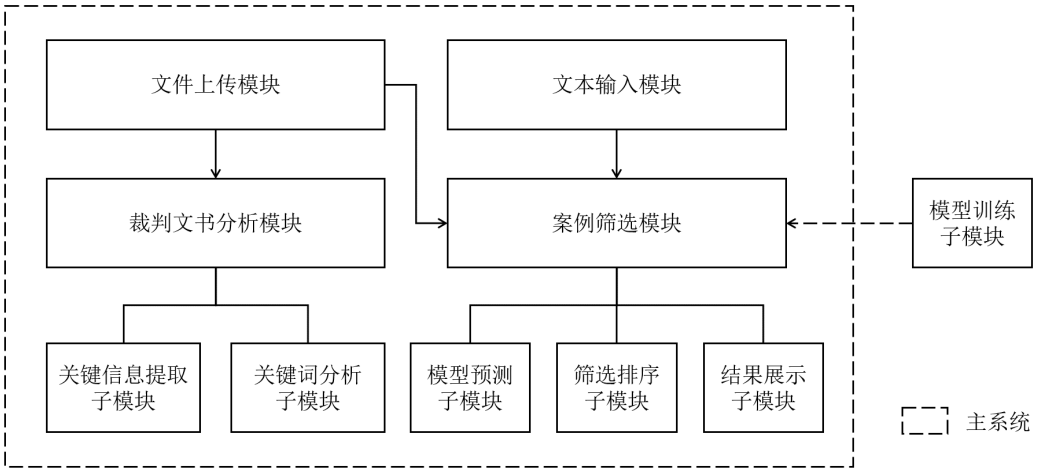

图 3.2: 司法案例筛选系统模块图

于裁判文书大多都是用户由各种途径获得的文本文件，因而可以采用直接上传的功能进行相似案例的筛选。上传的文件可传递给裁判文书分析模块进行解析，也可传递给案例筛选模块进行相似案例筛选。

裁判文书分析模块与文件上传功能紧密相关，能够对上传裁判文书进行分析并对关键信息进行展示。裁判文书是诉讼审判过程中的重要记录以及审判结果的唯一载体，因而篇幅相对而言会比较长。人工阅读需要花费一定的时间精力才能获取对应的关键信息。而裁判文书分析模块的设计目的就是提供裁判文书的快速解析服务。系统将帮助用户通过裁判文书分析模块提取出的关键信息和关键词对上传的裁判文书进行快速了解。裁判文书分析模块的结果展示页面作为文件上传成功的页面予以展示。

案例筛选模块接受“文本输入模块”和“文件上传模块”两个方向的信息流，在这一模块中统一转化成模块可分析的文本信息。案例筛选模块将调用文本分类模型对待分析文本进行标签预测，单一标签的筛选效果欠佳，系统将训练案例案由分类器和案例刑期分类器两个文本标签分类器。依据预测得到的标签从系统的案例库中筛选相似的案例文本，计算文本相似度以进一步排序，并将最后排序完成的结果进行展示。

对于案例筛选模块所需的文本标签预测模型，需要额外的模块加以训练（模型训练子模块）。这一模块所需的依赖环境复杂，并且所需文件占据空间较大。因此，这一模块与主系统隔离开，以降低主系统的复杂度。主系统仅需要调用这一子系统训练完成的模型及对应的支撑语料库即可。在实际部署时，仅部署主系统及训练好的模型即可，模型训练子模块无需部署。

3.3.2　系统开发设计

系统拟使用Python语言进行开发。根据上述的系统需求及模块划分，对系统开发架构设计如图 3.3所示：

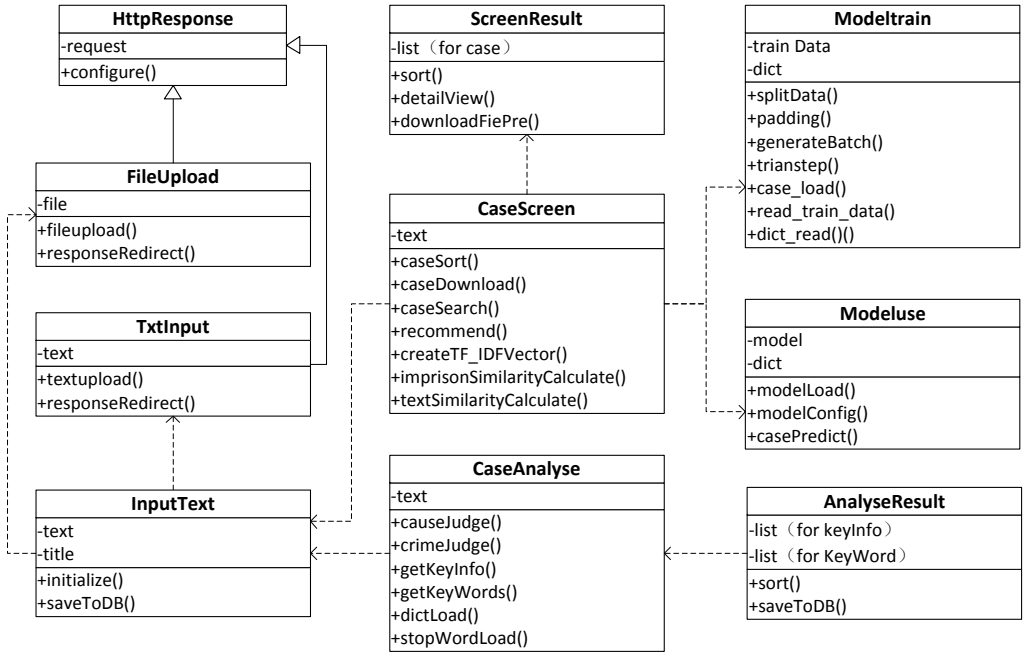

图 3.3: 司法案例筛选系统类图

对系统类图中主要类说明如下：

HttpResponse类：该类是本系统进行web传输时的基类，其主要功能是实现web通信中的各基础功能。封装完毕后供其他模块调用。

FileUpload类：该类继承HttpResponse类，主要实现文件上传功能，并负责对上传文件做合法性检测。在这一类中需设置上传页面的模板并调用相应变量进行渲染，并设置文件上传后的跳转逻辑。

TxtInput类：该类继承HttpResponse类，主要实现文本上传功能。与FileUpload类相似，同样需要设置模板，调用变量进行渲染并设置跳转逻辑。

CaseAnalyse类：该类主要实现裁判文书的解析工作，需要依赖FileUpload类或TxtInput类获得的数据进行解析。实际使用中，一般不对输入的文本做解析，但系统仍然预留相关的调用接口。该类中的成员函数实现较为复杂，包含对裁判文书的关键信息和关键词做提取，拟调用Python中的jieba等工具包，在后续章节中会以子模块实现的方式进一步展开论述。

Modeltrain类与Modeluse类：这两个类实现系统所需机器学习模型的训练和调用。具体包含对训练数据的处理，模型训练和调用等内容。该两类中的具体实现较为复杂，包含对机器学习框架的调用(fastText)，在后续章节中会以子模块实现的方式进一步展开论述。

CaseScreen类：该类实现案例筛选功能，需要依赖FileUpload类或TxtInput类获得的数据进行解析，依赖ModelUse类进行输入文本的处理。在CaseScreen类中将实现相似案例的筛选、排序、展示和下载等功能，拟调用Python中的genism等工具包,在后续章节中会以子模块实现的方式进一步展开论述。

上述是对司法案例系统开发的基本设计，后续章节将就开发过程中的实现细节及难点作进一步展开论述。

3.3.3　系统进程设计

根据上述需求及设计，本节将从用户使用及系统运行两个视角分析介绍系统进程设计。

用户视角：

根据用户需求分析，用户使用系统的主要目的是进行裁判文书的解析和相似案例的筛选。并且对于案例筛选功能，系统提供文本输入和文件上传两种交互手段。因而，用户使用系统的流程如图 3.4所示：

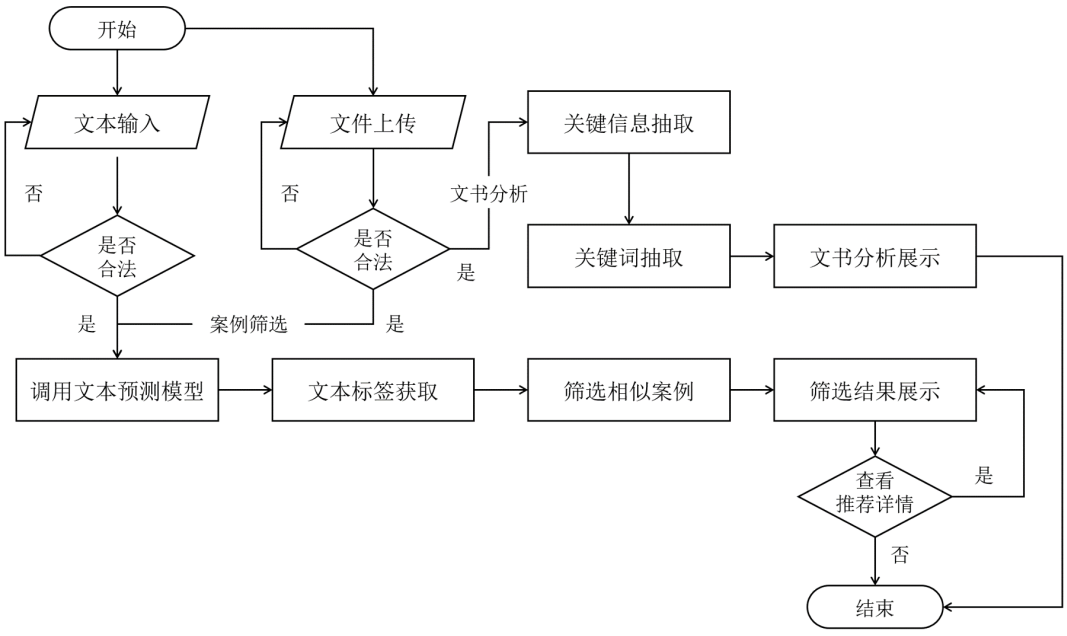

图 3.4: 司法案例筛选使用流程图

对用户使用流程如下说明：

1. 以文本输入为开始输入

以文本输入为开始输入，仅支持案例筛选功能。用户在文本输入界面输入待查询的文本。上传的文本会经过合法性检验（主要是对文本长度和字符类型的检验）。通过合法性检验的文本进入案例筛选流程，调用文本预测模型对文本进行处理，并进一步筛选出相似案例。筛选结果展示界面将展示输入文本原文及相似案例摘要信息，并提供下载链接。相似案例摘要信息可查看详情，若选择查看推荐案例详情，系统在显示详情文本的同时，将选择的案例作为输入文本再次进行案例筛选，此时可查看以被选择案例作为输入的筛选结果。

当用户查找到相似案例时，结束使用流程。

2. 以文件上传为开始输入

以文件上传为开始输入，支持案例筛选功能和裁判文书分析功能。执行案例筛选功能与以文本为开始输入时流程相同，不另作说明。以文件上传为开始输入执行裁判文书分析功能流程如下：上传的文件会经过合法性检验，不合规的文件无法分析。通过合法性检验的文件进入解析流程，展示上传裁判文书的文书类型、判决法院、涉及法律、案例案由等要素并结束裁判文书分析流程。

系统视角：

系统通过统一的前端界面与用户进行交互，在处理用户不同类别的请求时内部有不同的时序。系统在调用裁判文书分析功能时，除了解析并返回用户的请求外，同时会将文件和分析的结果进行存储。执行此功能时，系统对应的时序图如图 3.5所示：

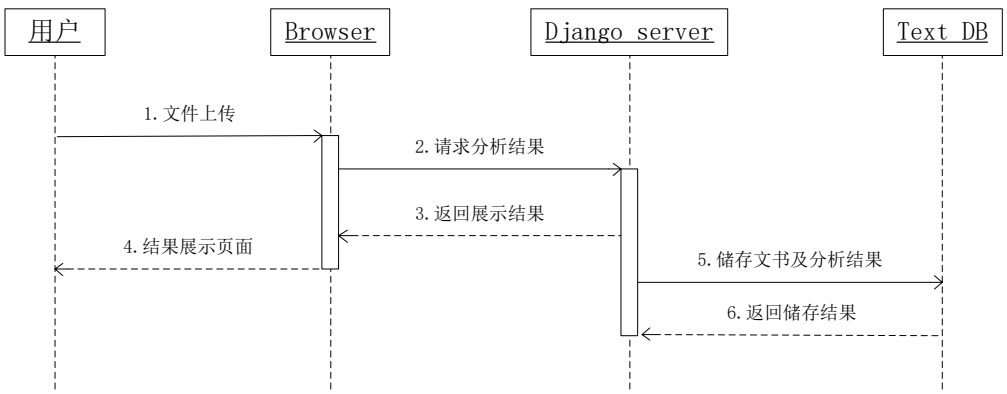

图 3.5: 裁判文书分析时序图

系统在调用案例筛选功能时，会向数据库发出检索请求，并对查询的结果进行筛选选排序，生成摘要结果并返回。执行此功能时，系统对应的时序图如图 3.6所示：

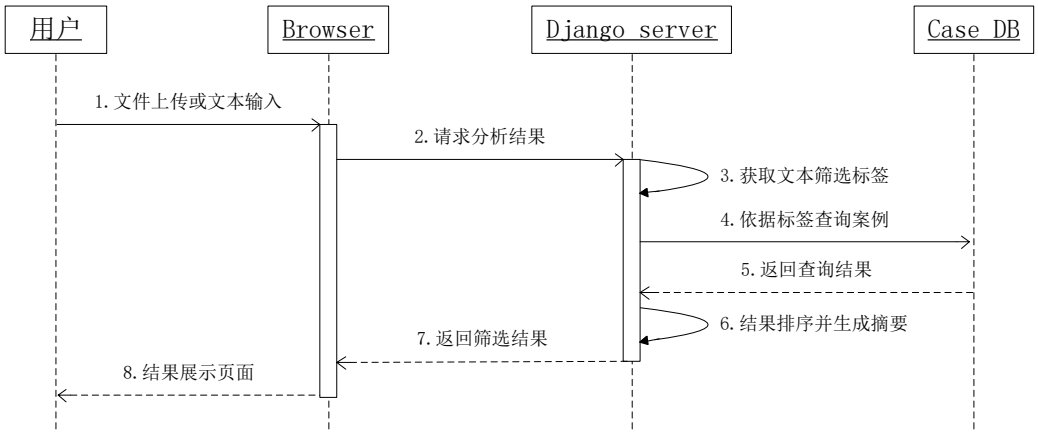

图 3.6: 案例筛选时序图

3.3.4　系统部署设计

系统整体部署设计如图 3.7 所示。用户通过用户机上的浏览器通过HTTP请求访问网站，由Web服务器做出页面响应。Web服务器（可选IIS，Nginx等）中由Django解析请求，并调用对应的Python应用程序进行处理。同时，相关的数据存储在Sqlite数据库中，由相关的Python程序进行连接和调用。

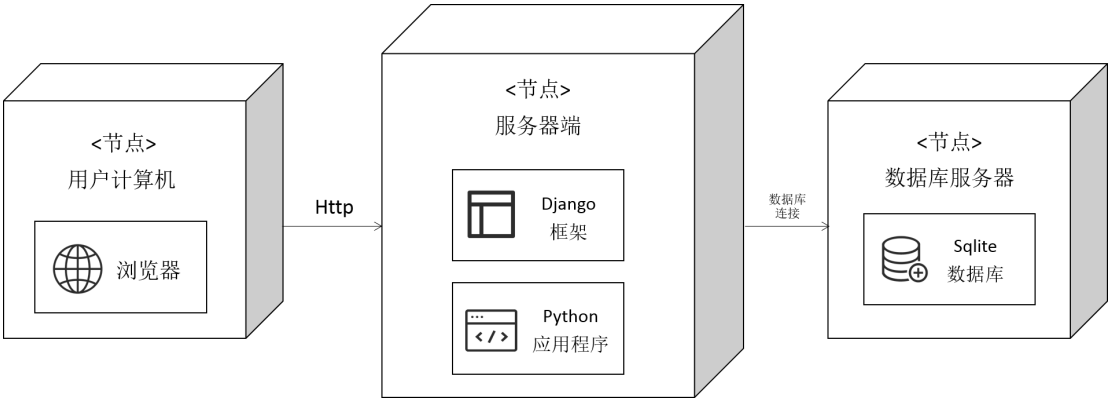

图 3.7: 司法案例筛选系统部署图

3.4　本章小结

本章首先对系统进行总体规划，进一步分析系统存在的功能性需求以及非功能性需求，并且通过用例描述表进行了总结和归纳。随后，通过模块图以系统使用流程图的形式构建起系统的概要设计，将系统主要模块以及模块之间的关系理清。最后，进行了系统的开发设计和运行流程设计，为下一章中叙述系统中个模块的详细设计和实现作铺垫。

第四章　系统详细设计及实现

根据第三章给出的司法案例筛选系统需求分析和架构设计，本章将利用Django及fastText等框架对各个模块进行详细设计，根据各个模块的特点给出对应的组件图、流程图，并对模块实现中的要点予以说明。

4.1　文本输入及文件上传模块设计及实现

文本输入模块和文件上传模块是系统用于和用户交互的主要模块。对应功能采用Django框架中的自定义视图函数实现。由于文本输入和文件上传的的处理过程较为相似，故在此统一进行说明。对用户输入的文本或上传文件处理的流程如图 4.1所示:

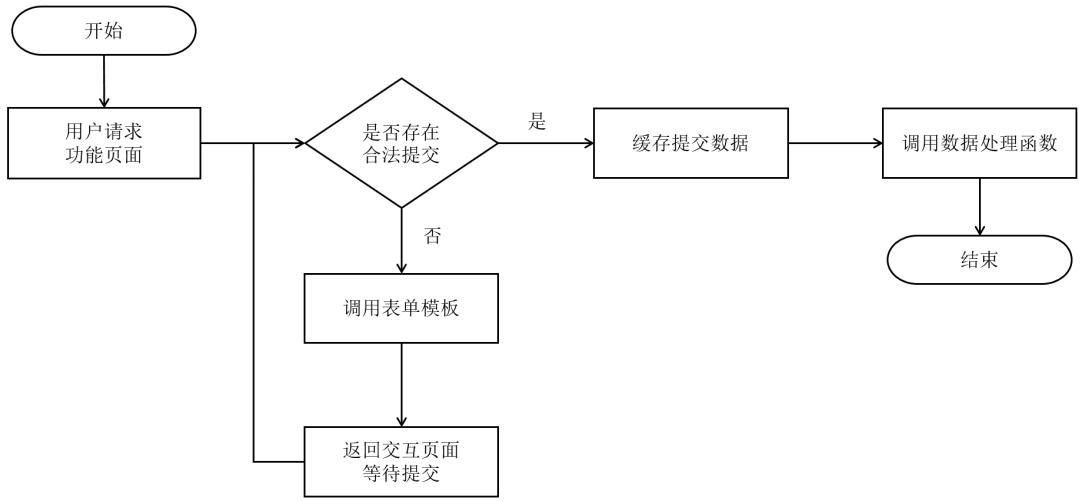

图 4.1: 表单交互处理流程图

首先，在用户请求后返回对应的交互表单，并判断用户是否提交。用户提交表单之后，系统保存表单内容至缓存，并将表单内容以参数形式传递给相关的文本处理函数进行处理，实现交互过程。其中主要的技术考量点如下：

1. 交互方式选择

在进行文本输入及文件上传时，需要有相应的交互方式将前端获取的文本传至后端的服务器。HTTP协议中定义了“请求-恢复”的工作方式。当用户由前端发送请求时，可以连通附件的数据一起发送。在这种情况下服务器只需解析请求，即可获得用户传送的内容。此时需要设计对应的HTML表单。常用表

单交互方法有GET和POST两种。GET 方法直接将参数和url一起发送,而POST方法是通过request body传递参数。鉴于上传的文本长度可能较长，且GET请求是在url中直接传送的参数，长度受限制。因而，采用POST方法进行表单交互。

2. Django对POST类型表单特有的防护机制

根据所选定的表单交互方法POST，在前端页面设计对应的交互表单（对于文本输入模块，包含一个多行文本框及提交按钮；对于文件上传模块，包含一个文件查看器及提交按钮）。值得注意的是，在Django中，直接使用POST方法进行表单交互时可能会产生错误。原因是Django有csrf防护机制，这是一种防止跨站请求伪造的手段。因而，在实际设计表单时，需要加入对应的token以应对这一防护机制 [40]。在具体表单的设计中，表单的首行代码需设置为“{% csrf_token %}”，以通过防护验证。

实现了上述统一流程后，系统将根据用户文本输入或文件上传两类不同的请求，调用对应的模板及表单。实现用户与系统间的交互。

4.2　裁判文书分析模块的详细设计与实现

4.2.1　裁判文书分析模块结构

裁判文书分析模块是系统中重要的模块之一，这一模块需要分析上传的裁判文书数据，并给出裁判文书中的关键信息。部分裁判文书的篇幅较长，直接进行阅读需要花费大量的时间精力。裁判文书分析模块旨在通过计算机分析的方式提取部分关键信息，替代人工阅读及关键字词的查找工作。一般而言，阅读者对裁判文书的基本了解需包含以下方面内容，如表 4.1所示:

表 4.1: 裁判文书分析要素特征表

要素名称	可能的取值
文书类型	6种，分别为判决书、裁定书、调解书、决定书、通知书和令
判决法院	有限个，但这一字段裁判文书中固定在一定位置出现，可提取
涉及法律	96种，目前我国法律总数量为96部
案由	841种，《民事案件案由规定》案由规定总数为841种
罪名	420种，《中华人民共和国刑法修正案》总计增补罪名420种
关键词	不存在准确取值，需要依靠算法提取

由上表可知，除关键词外的要素均有明确取值。并且取值的范围有限。因而对于这一类信息的提取可通过查询数据库或词典的方式，转化为对比分析的

任务进而实现提取。关键词提取需要通过算法计算获得结果，需要独立的算法支持。因而，裁判文书分析模块内部结构如下图 4.2所示：

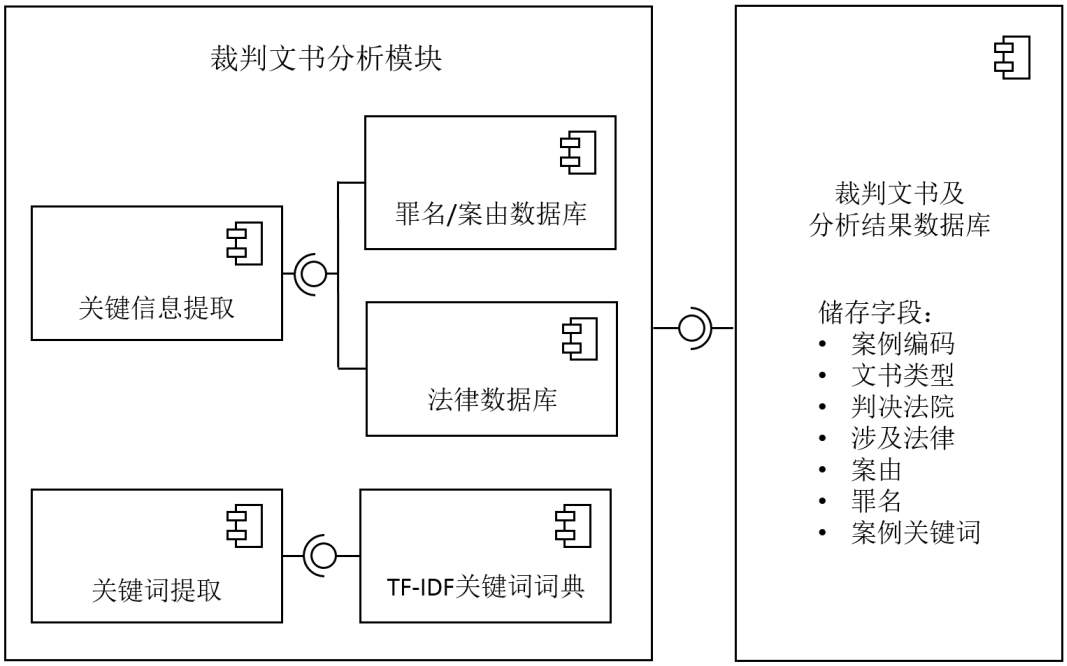

图 4.2: 裁判文书分析模块组件图

其中各部分说明如下：

1. 关键信息提取子模块：这一子模块对可通过查询数据库或词典提取的字段进行处理，包括文书类型、判决法院、涉及法律、案由或罪名。这一子模块直接处理裁判文书信息。

2. 关键词提取子模块：这一子模块对无法通过查询数据库或词典提取的字段进行处理，主要用于提取关键词。通过TF-IDF算法对进行分词后的裁判文书进行处理。

3. 罪名/案由数据库：用于支持罪名/案由提取。对于刑事类裁判文书，所有案件会依据相关法律归类于某一罪名下。对于民事类裁判文书，所有案例会依据相关法律归类于某一案由下。因此，通过罪名/案由数据库的构建，收录罪名和案由的全集，可对任意裁判文书进行罪名或案由的归类。

4. 法律数据库：用于支持涉及法律提取。对于任意裁判文书，相应的判决一定需要依据某一部具体的法律。因此，通过法律数据库的构建，收录审判依据法律的全集，可对任意裁判文书进行法律的归类。

5. TF-IDF关键词词典：用于获取关键词逆词频（IDF）。采用TF-IDF算法

时，统计文档中出现词语的逆词频（IDF）时需要对所有收录文档进行统计，由相关公式计算获得逆词频（IDF）值。由于词语的逆词频值相对固定，在初次构建语料库时会计算所有词的逆词频（IDF），并以词典的形式保存。在进行关键词提取时，无需再次计算，直接通过在词典中查找的办法获取逆词频值。

6. 裁判文书及分析结果数据库：用于储存已分析的裁判文书及结果。在处理文书前先查询是否在数据库中。对于数据库中处理过的裁判文书,不再次进行解析，直接调用分析结果。

4.2.2　关键信息提取子模块具体实现

关键信息提取子模块需要提取的字段包括文书类型、判决法院、涉及法律、案由或罪名。这些关键信息字段均在有限集合内取值。因此可以将待处理的裁判文书文本进行分词后视作一个词及短语的集合，这一信息与关键信息字段集合的交集，即为需要提取的关键信息。裁判文书由于其行文严谨规范，一般仅会出现与这一文书案例记载相关的关键字段。在少数情节复杂的案件中可能存在数罪并罚的情形，但同样适合于取并集的解决办法。

特别的，对于文书类型和判决法院字段，因为其在裁判文书较为固定的位置出现，可采用正则表达式直接提取。

对于涉及法律、案由或罪名，则需要构建可能的取值数据库。本文在网络及相关法律专业数据库上进行检索，并采取正则表达式和人工筛查等辅助手段建立数据库。其中法律数据库收录我国法律96部法律的具体名称；罪名/案由数据库包含《民事案件案由规定》中总计收录的841种案由 [41]和《中华人民共和国刑法修正案》收录的420种罪名 [42]。如图 4.3所示:

罪名清单截图

410	非法出卖、转让武器装备罪
411	遗弃武器装备罪
412	遗失武器装备罪
413	擅自出卖、转让军队房地产罪
414	虐待部属罪
415	遗弃伤病军人罪
416	战时拒不救治伤病军人罪
417	战时残害居民、掠夺居民财物罪
418	私放俘虏罪
419	虐待俘虏罪
420	危险驾驶罪

案由清单截图

831	申请认可和执行香港特别行政区仲裁裁决
832	申请认可和执行澳门特别行政区法院民事判决
833	申请认可和执行澳门特别行政区仲裁裁决
834	申请认可和执行台湾地区法院民事判决
835	申请认可和执行台湾地区仲裁裁决
836	申请承认和执行外国仲裁裁决
837	执行异议之诉
838	案外人执行异议之诉
839	申请执行人执行异议之诉
840	执行分配方案异议之诉
841	撤销权纠纷

法律清单截图

86	中华人民共和国农业机械化促进法
87	中华人民共和国动物防疫法
88	中华人民共和国反垄断法
89	劳动法
90	中华人民共和国就业促进法
91	中华人民共和国劳动合同法
92	诉讼及非诉讼程序法
93	中华人民共和国民事诉讼法
94	中华人民共和国仲裁法
95	中华人民共和国刑事诉讼法
96	中华人民共和国行政诉讼法

图 4.3: 法律数据库及罪名/案由数据库截图

完成关键信息字段取值范围数据库（法律数据库、罪名/案由数据库）收集后，即可通过比对数据库中记录提取关键信息。关键信息提取子模块获取待分

析的文本后，遍历数据库中的字段，检查其是否在待分析文本内存在匹配。若存在匹配，则该字段即为待分析文本需要提取的关键信息。具体流程如下图4.4所示：

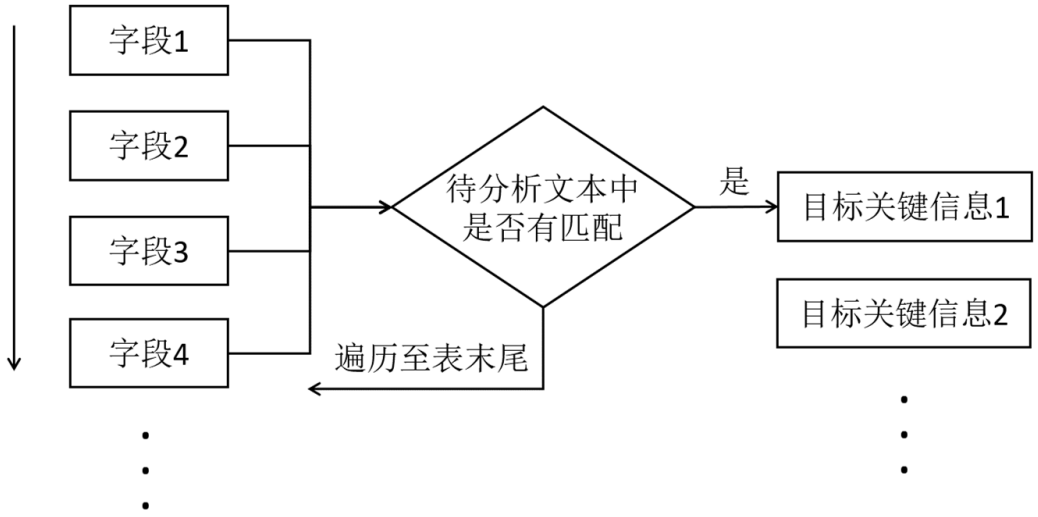

图 4.4: 关键信息匹配流程图

在具体实现中，匹配关键信息需遍历循环至数据库的末尾，没有采用在获取匹配后退出循环的方式。其原因是部分案例可能出现数罪并发的情况。在存在数罪并发的情况下，所有案例实际匹配字段数无法确定，因而只能采取遍历数据库的方式才能确保没有遗漏。

4.2.3　关键词提取子模块具体实现

关键词提取子模块需要提取裁判文书中具有一定代表性的关键词。关键词提取有诸多方法。根据上文所述的关键词提取需求，主要目的为帮助用户理解当前裁判文书，应当以当前裁判文书为处理核心。因而本文中的关键词提取不适宜借助其他外部信息，而应当从文本自身的特征角度出发，通过无监督的文本关键词提取手段获得文本的关键词。

TF-IDF是一种从文本自身统计特征出发，利用文本中所有词的词频信息提取文档中原有的词作为关键词的方法。TF-IDF算法以词频（TF）这一文本中最为基础且原始的统计信息为基础依据，一个词在文本中出现的次数越多，则代表这个词越能够带代表整段文本。尽管依靠词频会受到部分虚词的干扰，但加载停止词表后这一现象能大幅改善。逆词频（IDF）是语料库所有文档数量与包含关键词文档数量的对数值。用于表达对应词在整体语料库中出现的频繁程

度，其频繁程度越低，IDF值越高。具体的公式如下：

$$TF-IDF(x) = TF(x) \cdot IDF(x)$$

其中词频TF按如下定义实现：

$$TF(x) = \frac{F(x)}{F}$$

其中逆词频IDF按如下定义实现：

$$IDF(x) = log\frac{N+1}{N(x)+1}$$

在上述公式中，F代表文档中总的词数，F(x)代表统计词出现的词数。N代表语料库中文档总数，N(x)代表有统计词出现的文档数量。在逆词频IDF的实现时，通常在分式的分子分母同时加1做平滑，以预防分母为0的情形。

在使用TF-IDF算法时，首先需要对文本进行分词。在初步掌握分词原理后，本文采用jieba分析工具进行分词。TF-IDF算法首先统计词频，并且进一步通过TF-IDF关键词词典查找对词的逆词频（IDF）获得TF-IDF值。具体流程如下图 4.5所示：

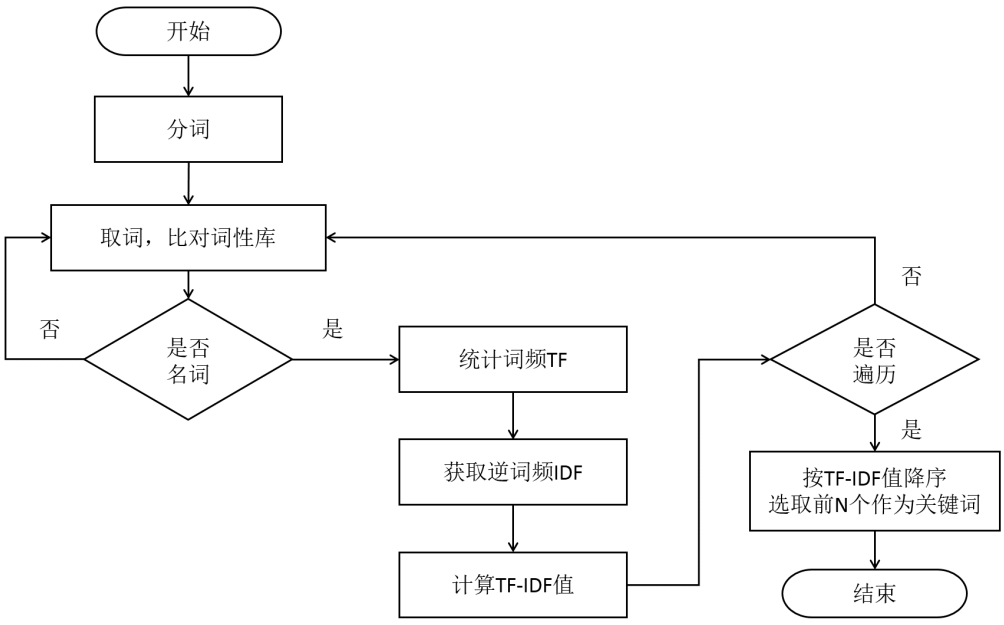

图 4.5: TF-IDF关键词获取流程图

关键词具体选取排序靠前的10-15个为宜 [43]。选取关键词数过少，对要素覆盖程度不足；选取关键词数过多，排位较低的关键词相关性不足。

在基础的TF-IDF算法之下，本文在具体实现中还采用了以下手段进一步优化改进关键词提取的效果：

（1）加载语义停止词。所谓停止词，是指“啊”，“的”，“呀”之类的虚词和语气词。这类词几乎在所有的文档中都有所出现，词频通常都不低。通常这类词的IDF也非常低，但是由于虚词和语气词几乎不表意，直接将这类词定位停止词。遇到停止词时，直接跳过不予处理。

（2）加载法律停止词。与语义停止词类似，司法领域专有的名词例如“公诉人”，“被告方”等，是案例中常见词汇，不能够刻画案件的特征信息，故也作为停止词处理。采用这一方法需要搜集并维护专门的法律专有名词表。

（3）词性制约。在司法文本中，案件特征一般由名词体现。其他如动词“实施”，形容词“严重”等出现频率也很高，占据关键词排行的前列。提取出的关键词还会出现一连串的动作，这样的关键词提取对案情的表述有限。将提取关键词的词性限制为名词，能够获得更好的提取效果 [44]。

4.2.4　裁判文书及分析结果存储功能具体实现

实现了裁判文书的所有分析工作后，需要对裁判文书原文及分析结果进行存储。需要储存的字段如下表 4.2所示：

表 4.2: 预储存字段表

字段名称	字段类型
文书类型	VARCHAR
判决法院	VARCHAR
案由	VARCHAR
罪名	VARCHAR
关键词	TEXT
文书内容	TEXT

储存内容主要以文本材料为主，对应的裁判文书并不存在直接可用的主键。在数据库设计中常有的主键包括自增编号、联合主键、GUID等方法。这些方法均能够生成唯一的主键。然而实际在使用数据库时，主要遇到的场景是通过裁判文书内容对数据库进行检索，查找是否有匹配案例。因此适宜围绕文书内容形成主键，方便检索使用。

MD5算法可以对任意文件生成信息摘要 [45]。通过MD5算法对文书内容进行计算，以生成MD5码作为储存主键，既可以解决记录主键的唯一性问题，也

可方便后续依据文书内容对数据库进行的检索（当对新输入的文书内容进行分析时，先计算其MD5码判断其是否在数据库中。若存在直接调用已分析的数据返回。避免以大段文书内容做查询）。

综上，在具体实现中建立数据库储存的字段如下表 4.3所示：

表 4.3: 实际储存字段表

字段名称	字段类型
MD5码	VARCHAR
文书类型	VARCHAR
判决法院	VARCHAR
案由	VARCHAR
罪名	VARCHAR
关键词	TEXT
文书内容	TEXT

实现裁判文书及分析结果存储的意义有以下两点：

1.进行存储后，再次遇到相同案例能够直接调用结果。

2.通过文档积累，进一步优化逆词频IDF的值，提升关键词提取效果。

4.3　案例筛选模块的详细设计与实现

4.3.1　案例筛选模块结构

案例筛选模块是系统核心的模块，实现相似案例筛选的主体功能。在这一模块中接收文本输入和文件上传模块获取的文本，根据机器学习模型获取文本特征标签，并由此筛选出相似案例并展示给用户。

这一模块总计包含4个子模块：模型训练子模块、模型预测子模块、筛选排序子模块和结果展示子模块。其中模型训练子模块与主系统完全分隔开。其与主系统的联系在于主系统会调用模型训练子模块训练出的模型文件。

模型训练子模块与主系统不同步工作，当数据集获得更新时，可单独使用模型训练子模块对模型进行重新训练，训练后将更新主系统调用的模型文件，确保主系统采用的模型不断迭代，确保系统获得最新语料训练出的模型。

模块的具体结构如图 4.6所示，对子模块及相关组件的说明如下：

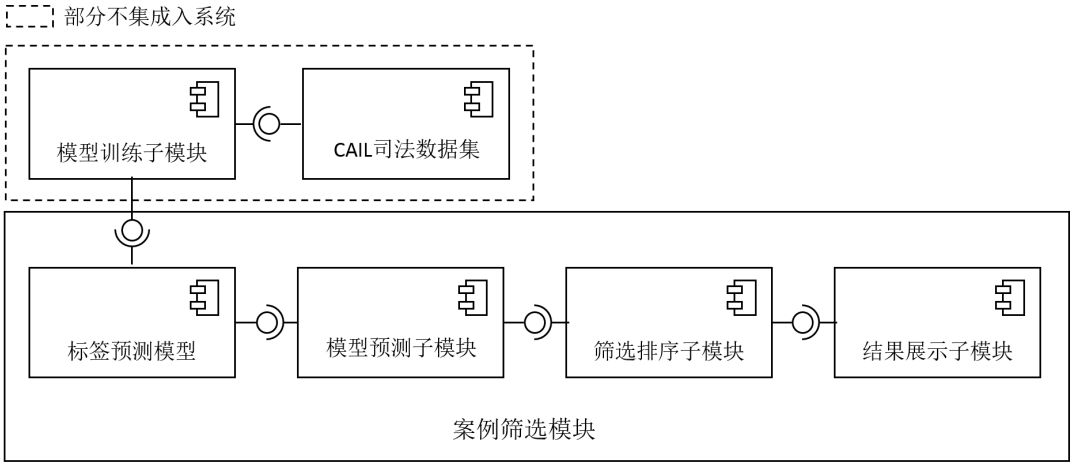

图 4.6: 案例筛选模块组件图

1. 模型训练子模块

模型训练子模块独立于主系统，通过fastText这一机器学习框架训练文本分类模型。根据需求设计，模型训练子模块需要训练案由分类器和刑期分类器。不同分类器训练的标签及原始数据相互独立，需要采取不同的训练数据。模型训练子模块输入为训练数据集，输出为分类模型，供预测子模块调用。

2. CAIL司法数据集

CAIL司法数据集用于模型训练子模块的训练。这一数据集中的数据将被整理为模型训练子模块可用的格式进行训练。数据集的具体内容和用法会在下文中详细介绍。

3. 模型预测子模块

模型预测子模块调用文本分类模型及其对应支撑的语料库。根据需求设计，模型预测子模块输入为文本输入和文件上传模块获取的待分析的文本，输出为待分析文本对应的标签。与模型训练子模块对应，模型预测子模块将调用案由分类器和刑期分类器对文本进行分类，并将获得的分类标签提供给下一子模块作为筛选依据。

4. 筛选排序子模块

筛选排序子模块的输入包含待分析的文本、文本对应的案由标签及刑期标签，输出是筛选出的相似案例列表。

在筛选排序子模块中，需要先根据标签筛选出一定数量的相似案例，并进一步通过标签信息和文本相似度对筛选出的案例进行进一步的筛选及排序，最终返回筛选出的备选案例列表。

5. 结果展示子模块

结果展示子模块的输入包含输入的文本信息及筛选出的相似案例列表，输出是向用户展示的界面。在结果展示子模块中还包含对用户在查看展示时相关拓展操作的实现，包括“详情查看”、“文件下载”等功能。

4.3.2　模型训练数据集及数据预处理

本文需要相应的数据集用于预测模型的训练。目前公开的文本数据集中，以通用的文本分类数据集为主。这类数据集通常是由新闻等语料收集而成。例如THUNews中文文本分类数据集，其中文本语料的题材来自各个行业和方向，仅有部分文本是和司法领域相关的。针对司法领域内部，包含具体罪名的细分数据集，目前在公开领域数量稀少。

优质的司法数据集，应当包含完整的司法事实，并对这些司法事实所对应的司法要素作清楚明确的标记。获得优质的标记数据后，即可针对相应的司法要素形成训练数据，训练模型和以解决对应的问题。

近年来，深度学习技术的不断发展，带动了整个自然语言处理技术的进步。作为人工智能的重要分支，自然语言处理技术在司法领域也开始有一定的落地应用。法律智能就是其中关键的一个环节。这一环节，受到了学术界、产业界及司法界三方共同的支持。在最高人民法院信息中心的牵头下，多方合作共同于2018年举办了中国“法研杯”法律智能挑战赛，也称为CAIL2018 [46]。

这一比赛首先对法律智能有了一个清晰具体的定义。法律智能是旨在使得计算机能够通过对应的人工智能程序处理分析法律文本，并从某种程度上达到类似人类“理解”的效果的计算机程序。在具体法律事务方面，要求法律智能能够依据案件事实完成各项司法要素的预测并给出相关定量分析，包含罚金刑期预测，进行具体的法律条文推荐等等 [47]。

法律智能的目标，不是采用机器判案，而是通过法律智能更好地帮助律师、法官等司法从业人士高效地进行辩护审判等工作，使他们能够从繁琐的案例检索和收集，法条查阅和比对中抽身出来，将时间和精力都集中在需要通过从业人员的司法意识和主观能动性解决的事情上 [48]。

在法律智能的定义下，挑战赛提供了大量法律文书作为数据集。这些文书数据来自于“中国裁判文书网”。与通常公开的裁判文书不同，数据集中的每份数据是经过处理的，分为“案情描述”和“事实要素”两个部分。每个案件所涉及及的具体法律条文、被告人所受到判决的罪名，以及具体刑期的年份都有被清晰的封装为标签。在数量方面，目前的数据集一期包含近20万条记录，能够保证模型的基本训练。

数据集中的数据用JSON格式存储，每条记录均储存为一个字典，包含各个要素信息，具体字段的含义如下表 4.4所示。

表 4.4: 数据集字段含义表

要素名称	字段含义及解释
fact	事实描述
criminals	被告人，数据集中均只包含一个被告
punish_of_money	判决罚金
accusation	罪名，被指控的罪名
relevant_articles	案例相关法条
term_of_imprisonment	刑期具体格式
death_penalty	是否死刑
life_imprisonment	是否无期
imprisonment	有期徒刑刑期

通过标注信息和案件的事实描述，可以就某一具体的字段构建并训练基于fastText的机器学习模型，得到对应的案例标签分类器，进而用于案例筛选。本文需要构建的案由分类器和刑期分类器，具体对上述数据处理的预流程（以单条数据为例）如下图 4.7所示。

数据预处理过程中需要对‘accusation’字段和‘imprisonment’字段进行处理，对流程图的具体说明如下文所示。

1. 对‘accusation’字段的系列处理

‘accusation’字段记录了案件的罪名（对刑事案件）或案由（对民事案件），为表述方便在下文的叙述中对罪名和案由两者统称为案由。‘accusation’这一字段主要由文字短语组成，部分字段的表述较长，例如字段“隐匿、故意销毁会计凭证、会计帐簿、财务会计报告”。‘accusation’字段这一长短不一的文字字段不适宜作为文本分类时直接使用的标签。因此，需要对‘accusation’字段进行编码，引入字段‘num_accusation’。‘accusation’字段共有202个可能的取值，‘num_accusation’这一字段将‘accusation’字段映射为数字1-202。

在实际训练中，案由分类器训练数据以‘num_accusation’字段作为标签，以‘fact’字段作为描述输入，可以省去模型对标签的再次编码。

2. 对‘imprisonment’字段的系列处理

‘imprisonment’字段是‘term_of_imprisonment’中的子字段，记录了有期徒刑的刑期。有期徒刑的刑期一般单一罪行不高于15年，数罪并罚不超

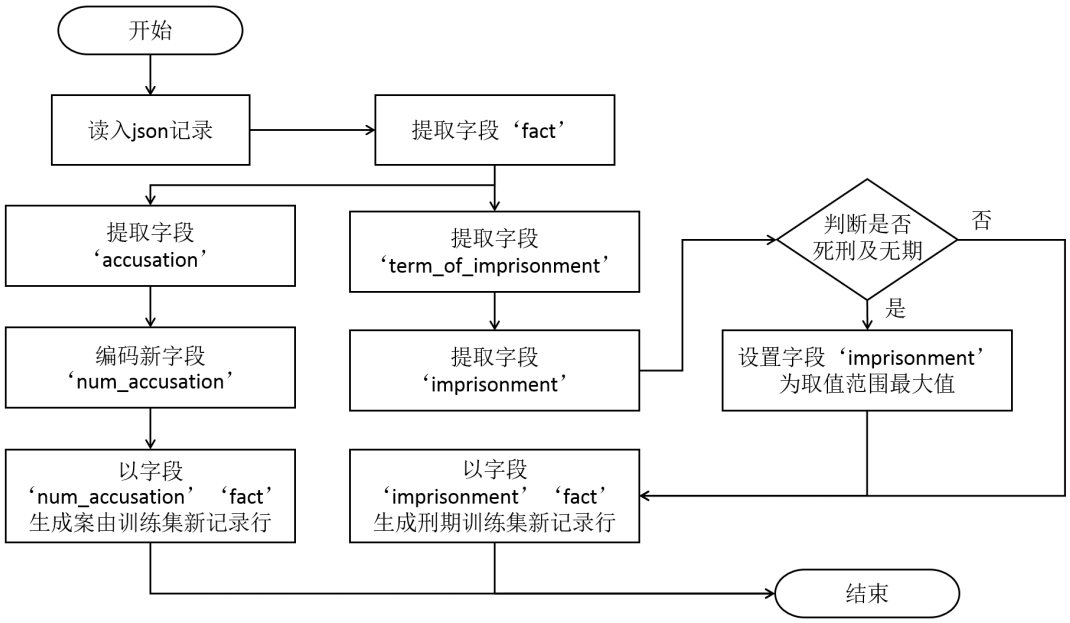

图 4.7: 数据预处理流程图

过25年。‘imprisonment’本身是取值为0-25的数字字段，适合作为训练标签。但并非所有的案例都有合法的‘imprisonment’取值，‘term_of_imprisonment’字段中的‘death_penalty’字段和‘life_imprisonment’分别标记了案例最终的判决是否是死刑和是否是无期徒刑。在这两个标签有有效取值下，‘imprisonment’字段的值设置为-1。作为纯粹的标签分类问题，取值为-1是可以接受的。但是本文需求的刑期分类器旨在通过分类结果案件的严重程度，需要达到分类结果标签值越高证明案件判罚的结果越严重的目的。因此，无期徒刑和死刑刑期取值设置为-1是不符合分类器意图的，在数据预处理时，将无期徒刑和死刑刑期取值设置为‘imprisonment’取值的最大值，以反映这类案例的判决严重程度。

在数据预处理阶段，对原始数据集的json格式的每一条数据记录按照图 4.7所示的流程进行处理，可以得到两条新的记录行。完成对整个数据集的处理后，可得到两个新的数据集，分别是以‘fact’字段作为输入，以‘num_accusation’字段作为标签的案由训练集和以‘fact’字段作为输入，以‘imprisonment’字段作为标签的刑期训练集。这两个训练集构建完备后，将能够直接被模型训练子模块调用进行训练。

4.3.3　模型训练子模块具体实现

本文采取fastText作为案例筛选时的分类模型。

fastText是专门用于文本分类的开源框架。fastText的结构相对深度学习模型更为简单，因而在训练和调用的速度上优势明显，并且分类效果优良（可参考第二章的相关介绍）。

fastText单层神经网络的结构，使得文本结构信息的丢失严重。在实际训练中通过引入N-gram向量后（N取值设定为2），保留了词与词之间的关系，使得结构信息也能用于文本分类的处理，一定程度上弥补了为追求速度设计单层神经的缺陷，获得更优的分类效果。fastText所需要训练的模型是均是三层的神经网络。训练模型时，需要对预处理完成后的数据进行进一步处理，包括词典生成，数据集划分等步骤，具体流程如下图 4.8所示：

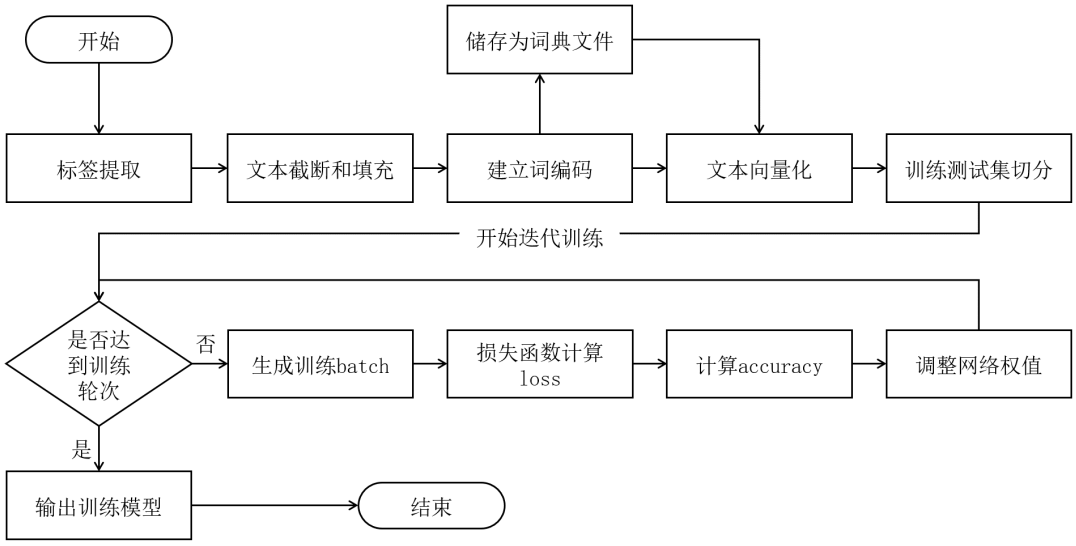

图 4.8: 模型训练流程图

模型训练中的各个步骤简述如下：

1. 标签提取-文本截断和填充

模型训练首先读入预处理后的数据，将每条记录截断为数据（简记为x）和标签（简记为y）部分，形成数据集合X和标签集合Y，这一部分称之为“标签提取”。对于数据集合X，模型训练中要求输入的数据长度固定。因此，对于长度超过规定限度的数据，直接进行定长截断；对于长度不足的数据，调用padding函数对其进行空值填充。经过标签提取和文本截断和填充步骤后，将预处理后的数据转化为标签集合Y长度固定的数据集合X。

2. 建立词编码-文本向量化

对于文本数据，在进行处理时需要进一步将其编码化。建立词编码-文本向量化这一过程，建立了词与编码之间的映射。在这一环节中，将所有文本中的

词进行编码，并将对应的映射关系储存为词典文件。词典设置数据上限，词典以外的词将被赋予空值。经过这一过程后，固定长度的文本将转化为数值向量用于后续计算。

3. 储存为词典文件

对于建立词编码-文本向量化这一映射过程中所成的词典文件，需要进行文件化存储方便后续的调用。这一映射关系是最终模型的一部分，在调用模型预测时，首先需要调用这一词典文件对待预测的文本进行向量化，随后才能进行后续的计算。

4. 训练集测试集切分

完成数据集的向量化后，需要把数据集按照一定的比例切分为训练集、测试集和验证集。训练集用于训练网络参数，测试集用于测试训练结果（计算Test dataset accuracy），验证集用于辅助参数的确定（计算Evaluation）。经过这一过程后，数据集被分为三部分供后续过程调用，模型的迭代训练开始。

5. 生成训练batch

在代入数据训练时，无需每次都使用全量训练数据进行迭代。在训练集中随机抽取一部分作为当次的训练batch，同样能够获得近似的效果。在具体实现中每次训练时均调用Python包numpy中的函数random.permutation生成随机batch用于训练。

6. 损失函数计算loss

损失函数用于度量模型预测值与实际值的偏差，通过损失函数计算出的loss值反应了预测值与实际值的偏差。通过不断调节网络权重和偏差使得loss值不断减小是模型训练的优化目标。在实际实现中采用的损失函数是softmax_cross_entropy,由Python包TensorFlow中的相关函数实现。具体的实现流程主要分为两部分：

(1) softmax过程 [49]：由于神经网络输出的神经元个数与要分类的类别数N相等，因而需要先通过softmax函数对输出的各类别分量进行归一化。对于长度为N的输出向量，第i分类的softmax值计算公式如下：

$$softmax_i = \frac{exp(x_i)}{\sum_{j=1}^{N} exp(x_j)}$$

(2) 交叉熵损失函数计算：在经过softmax函数的处理后，使用交叉熵(cross entropy) [50]对神经网络输出预测的向量与样本所属的真实分类进行loss计算。样本真实分类的向量是one-hot形式的，即形似[0,0,0,1,0,……]，长度同样为N

（例如，若样本属于第4分类，则向量中的第4位为1，其余位为0）。以y'表示训练标签中的值，y表示预测向量softmx归一化输出的对应分量，向量中每一项的交叉熵可按如下公式计算：

$$H_{y'}(y) = -\sum_{i} y_i' log(y_i)$$

(3) loss计算：最终loss的值是预测输出向量与样本真实分类向量由(2)式中求得交叉熵向量的平均值，即为：

$$loss = \frac{\sum_{j=1}^{N} H_j}{N}$$

7. 计算accuracy

完成loss的计算后，调用测试集计算当前网络分类的精确度。

8. 调整网络权值

定义和计算loss值后，需要采用反向传播算法 [51]调整网络权值。在实际使用中，调用Python包TensorFlow中的优化器AdamOptimizer实现这一功能，同时需要设定Optimizer的学习率。当模型按照上述流程迭代足够的训练轮次后，保存当前网络对应的权值和偏置输出训练模型，完成训练。

在上述过程中需要设置诸多训练的参数。包括输入文本长度（文本超长则截断，长度不足进行填补），模型词典大小等等。在训练中需要设置的参数、参数含义及设定值如下表 4.5所示：

表 4.5: 训练参数设置详情表

参数名称	含义及解释	设置值
sequence_length	文本长度，大于该长度截断，低于填充	300
num_classes:	分类数量上限	202
vocab_size	字典大小	5000
embedding_size	embedding词向量维度	300
batch_size	每次训练batch大小	50
num_epochs	epoch数目	10
train_test_dev_rate	训练集，测试集，验证集比例	[0.97 0.02 0.01]
learning_rate	学习率	0.003

本文需要构建案由和刑期分类器，因此需要训练两个模型。

根据上述流程分别采用事先预处理好的案由训练集和刑期训练集进行训练。完成训练后保存训练好的模型及进行文本向量化的映射词典，供模型预测子模块进行调用。

4.3.4　模型预测子模块具体实现

模型预测子模块输入为待分析文本，输出为分类标签。在文本处理方面模型预测子模块与模型训练子模块有诸多相似之处，具体的流程如图 4.9所示：

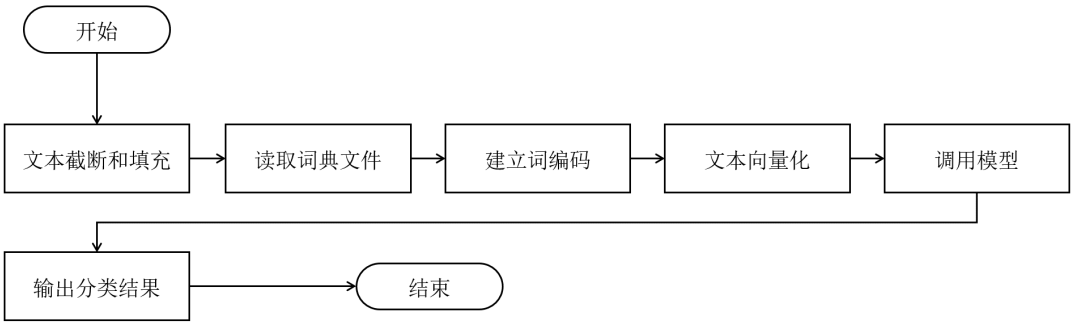

图 4.9: 预测文本标签流程图

文本截断和填充及文本向量化过程与模型训练子模块中的处理一致，唯一的区别在于调用模型预测子模块时不构建词典，而是调用事先保存的对应模型词典。并且对于不同模型，同一数据的形成词典也不相同（即使用同一数据集训练罪行预测模型和刑期预测模型，各模型形成的词典也不相同）。需要将当前模型以及字典文件的路径作为参数传给预测函数。

在实际实现中，本文构建的案由分类器和刑期分类器需要传递两组模型及的路径参数。由于模型训练后生成的模型及词典默认名称一致，因此两个模型及对应词典需要进行手动重命名，避免相互覆盖。

4.3.5　筛选排序子模块具体实现

筛选排序子模块，主要功能是依据待分析的文本及机器学习模型预测的标签从案例数据库中筛选出相似的案例，并对筛选出的结果进行排序。最终的筛选结果将以案例摘要信息列表的形式进行展示。

筛选结果中的案例数量与筛选条件相关。筛选条件越为严苛，符合条件的案例就越少。在实际应用中，设置相对固定的筛选条件会使得系统灵活性较差，使得部分案例无法获得有效的筛选结果。

为保障系统的流畅性和可用性，系统在筛选案例时会对筛选结果的数量进行控制：对于筛选结果较多的案例，择优展示（选取相似度高的展示）；对于筛选结果较少的案例，系统将进一步放宽检索条件直至获取可用的筛选结果。

对于筛选结果数量的具体设定，系统参考交互设计中的“7±2法则”[52]及席克定律 [53]。在“7±2法则”中指出，人类头脑能够较好的记忆5 - 9 项信息块，多于这一范围后，记忆便容易出错。这一法则在相互设计中被时常应用，通常各类应用一次可用的选项不会超过7 个。

进一步地，席克定律指出一个人面临的选择越多时，做出的决定的时间就越长，可用公式近似表示为（a,b为常数）：

$$T = a + b \cdot log_2(n)$$

这一定律在交互设计中意味着选项越多，用户做出选择的时间就越长。

筛选结果展示中，信息量不宜过低，展示数量应尽量多。但同时综合考虑“7±2法则”及席克定律，展示数量过多会使用户记忆负担过重且选择时间偏长。因而，系统将筛选展示的案例数量设置为7 。

在筛选过程中，首先通过先前构建的案由分类器和刑期分类器预测的标签确定检索条件，生成对应的检索语句，在案例数据库中进行检索。

具体流程及筛选条件的变更如图 4.10所示：

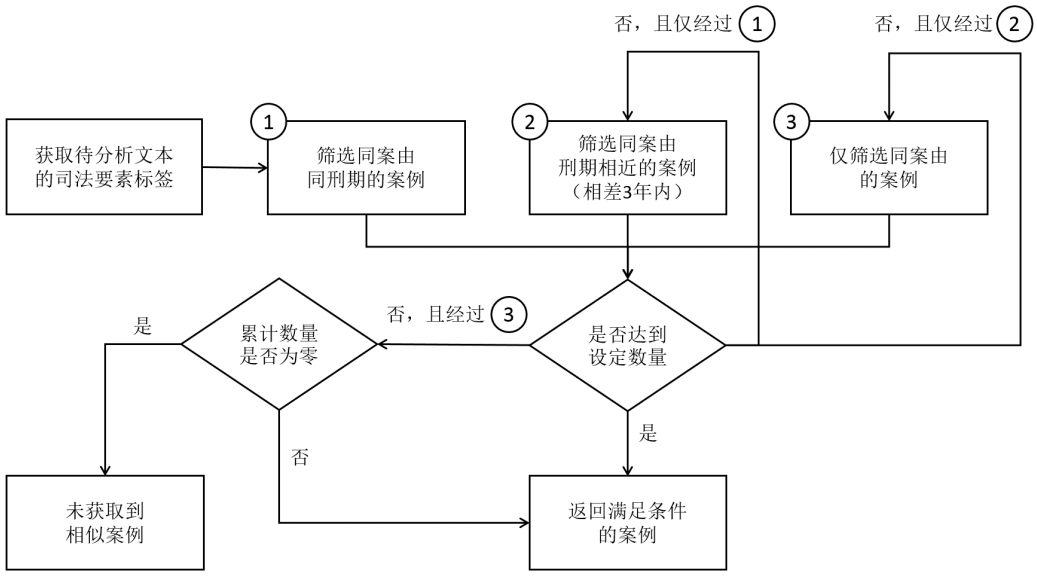

图 4.10: 文本筛选流程图

对筛选流程具体说明如下：

用于筛选的案由标签和刑期标签有着不同的特点。案由标签代表具体分类，标签值是离散的；而刑期标签代表刑期的具体数值，因而标签值实际是连续的整数。案由标签用于案例的类别划分，而刑期标签则一定程度上反应案例的严重程度。一般而言，统一案由下的案例都存在一定的相似性，故案由标签一致是筛选时的必要条件。刑期标签通过案例严重性的角度对案例间的相似性给予进一步的反映。在同案由案例数量较多的情况下，刑期标签能够帮助筛除大量的低相关度案例。对于大多数案例，通过同案由同刑期标签的筛选得到的备选案例是展示的第一选择。

对于部分案例，同案由同刑期的筛选标准过于严苛，导致筛选出的案例数量过少，此时将放宽检索要求。由于刑期标签是连续的整数，因此刑期标签的相近也代表备选案例与输入案例的相似性。在同案由同刑期的案例数量不足时，系统将刑期标签的要求放宽，设定为±3年。在这一条件下若仍然无法筛选到足够的案例，则将放弃刑期标签的制约，将案由标签相同的案例作为备选。

经过上述筛选过程，将获得备选案例列表。由于最终展示的案例数设置为不超过7个，在筛选环节中将对备选案例列表进行适当的截断。在筛选环节后将输出不超过14个备选案例列表（保留部分冗余案例在相似度排序阶段再淘汰）进行进一步的排序，最终选取相似度前7的案例进行展示。

在排序阶段，将进一步量化备选案例和输入案例之间的相似度，获得有序的备选案例列表。具体实现按照如下步骤进行：

1. 刑期相似度及文本相似度计算

以a表示输入案例的刑期，b表示备选案例的刑期，刑期相似度如下所示：

$$sim\_imprison = 1 - \frac{|a-b|}{a}$$

文本相似度采用基于TF-IDF向量的余弦相似度作为表示，通过文本层面词频的关系刻画案例间的相似度。计算文本相似度时，首先根据待计算文本构建TF-IDF向量。其中，TF-IDF向量长度为语料的词数，向量中的每一位代表一个词，每一位的值即为该词对应的TF-IDF值。

例如，对文本“abc”和文本“def”,以abcdef的顺序构建TF-IDF向量，则如下所示（设每个字母的TF-IDF值均为1）：

$$\text{”}abc\text{”} \Rightarrow [1,1,1,0,0,0]$$

$$\text{”}def\text{”} \Rightarrow [0,0,0,1,1,1]$$

对文本进行TF-IDF向量化后，即可计算文本相似度，以a表示输入案例向量，b表示备选案例向量，文本相似度下式所示：

$$sim\_text = \frac{< a, b >}{|a|\,|b|}$$

2. 案例排序

获得了刑期相似度和文本相似度后，即可对案例进行排序。刑期相似度直接由标签值计算获得，相比文本相似度更为准确。因而，以刑期相似度为第一关键字，文本相似度为第二关键字进行双变量降序排序，即先比较刑期相似度，在刑期相似度相同时比较文本相似度，确定案例顺序。

本文将采用genism工具包采用上述方法进行文本相似度的计算用于排序。完成排序后，选取相似度前7的案例生成备选案例列表，列表仅包含备选案例在案例数据库中的ID，案例题目、刑期相似度及文本相似度。

4.3.6　结果展示子模块具体实现

根据需求设计，结果展示子模块需要展示输入案例的原文及筛选出相似案例的列表，并实现详情查看和案例下载的功能，实现的主要工作如下：

1. 筛选结果展示

为确保筛选结果展示界面不显臃肿，展现筛选结果时不显示文本详情，仅显示备选案例标题，备选案例与输入案例的刑期相似性及文本相似性。更多详情信息需要用户点击查看。案例相似性和刑期相似性均为归一化数值，拟采用工具进行可视化显示。

系统采用前端实时绘制图标的形式显示案例的刑期相似度和文本相似度。调用ECharts图形库 [54]绘制堆积条形图描述相似度。刑期相似度与文本相似度采用独立的堆积条堆展示。每个堆积条堆由两个堆积条组成，堆积条的值之和为1。左侧堆积条值为相似度值，右侧堆积条值为（1-相似度）。

绘制效果如图 4.11：

图 4.11: 相似度可视化效果图

同时，对于每一个备选案例，系统生成查看详情链接和文本下载链接。用户可点击对应链接进行详情查看或案例文本下载。在备选案例展示列表的底部，系统生成批量下载链接，用户可以一次下载上述全部筛选结果案例。

2. 展示界面设计布局

根据系统需求对结果展示界面的布局设置如图 4.12：

图 4.12: 结果展示界面布局图

展示界面左侧区域展示当前输入的文本信息，右侧区域展示案例筛选的备选案例列表。通过这样的布局，既帮助用户将注意力集中在案例文本的阅读上，也兼顾了用户对筛选出相似案例的查阅。

用户点击查看详情后，备选案例详情将替代原文本显示在左侧区域，右侧区域也将替换为备选案例的筛选结果。用户可以在这一界面下进行连续的案例筛选，通过对备选案例的选择反馈当前对期待案例的偏好。

同时，利用界面上方的快捷检索框，用户可以在当前查看的文本中选择感兴趣的关键片段进行进一步的检索，通过减少无关输入的方式获得更精确的筛选结果。

3. 详情查看功能实现

当用户点击某一备选案例的详情查看链接时，系统将通过备选案例的ID在案例数据库中进行检索，获取备选案例内容文本。随后，并不直接显示在左侧

区域，而是以其作为输入文本再次进行案例筛选，完成筛选后更新页面。备选案例文本显示在页面左侧的文本详情展示区，右侧展示以选中备选案例作为输入的案例筛选结果。具体时序图如图 4.13：

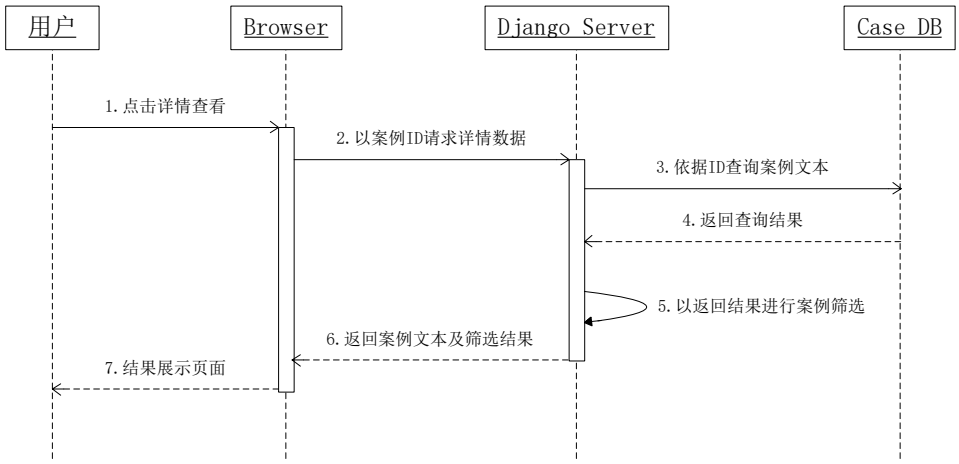

图 4.13: 详情查看功能时序图

3. 文件下载功能的实现

文件下载功能的实现与详情查看功能类似，故不另附时序图。当用户点击某一备选案例的详文件下载链接时，系统将通过备选案例的ID在案例数据库中进行检索，获取备选案例内容文本。随后，将其保存为文件返回文件链接至前端供用户下载。

特别地，用户点击筛选结果列表底部的“全部下载”链接时，系统将获取整个列表案例的ID，逐个查询并保存至同一文件并返回至前端作为下载源，实现批量下载功能。

4.4　系统测试

本章上述小节给出了司法案例筛选系统的具体设计及实现。本节将对实现的上述功能进行测试，并给出相应的测试用例和运行截图，主要内容如下：

1. 测试交互功能，包括系统是否能够正常接收上传文件及输入的文本。

2. 测试裁判文书分析功能，考察系统是否能准确解析出相应结果。

3. 测试案例筛选功能，考察系统是否能筛选出合适数量的相似案例。

4. 测试文本下载，备选案例详情查看等辅助功能。

4.4.1　系统测试环境

系统测试开发均在Window环境下进行，使用Window作为测试环境。

项目部署采用JetBrainsPyCharm自动部署Django项目，保障Django APP的稳定运行。测试与开发的环境详情如表 4.6所示：

表 4.6: 测试与开发环境详情表

环境名称	详情
系统环境	Window 8.1
数据库	Sqlite 3.0
开发测试软件	JetBrainsPyCharm Notepad++
硬件配置	4G 内存 128G存储空间
依赖环境版本	Python 3.0 Django 2.0 jieba 0.37 TensorFlow 1.41

4.4.2　系统测试过程及结果

1. 交互功能测试

交互功能测试中，包括文件上传测试及文本输入测试。文件上传具体测试用例如下表 4.7所示：

表 4.7: 文件上传测试用例表

测试用例编号	TestCase1
测试内容	用户上传欲分析的文本
测试功能	文件上传
测试步骤	1.访问web应用中文件上传对应的url
	2.选择想要上传的文件
	3.点击“提交”按钮，文本框内的文本上传服务器进行处理并对文件进行解析
预期结果	正常解析文件内容，文件缓存到指定目录下

系统实现的文件上传界面，包括文件选择按钮和上传提交按钮。布局如图4.14所示。测试时选择文件后，点击“上传”进行提交，文件上传前后，文件缓存文件夹情况如图 4.15所示。

图 4.14: 文件上传界面图

上传前：

上传后：

图 4.15: 缓存文件夹变化图

文件上传成功，系统成功将上传文本保存至缓存文件夹内，满足预定需求。对于文本输入功能测试流程与文件上传上述过程相似，故不另设测试用例。文本输入界面如如图 4.16所示。文本输入功能经测试，亦满足预定需求。

图 4.16: 文本输入界面图

2. 裁判文书文书分析功能测试

裁判文书分析功能测试，上传已知分析结果的裁判文书，考察系统是否能够正确给出分析结果。裁判文书分析功能具体测试用例如下表 4.8所示：

表 4.8: 裁判文书分析功能测试用例表

测试用例编号	TestCase2
测试内容	用户查看裁判文书分析结果
测试功能	裁判文书分析
测试步骤	1.访问web应用中裁判文书分析对应的url
	2.选定待分析的裁判文书
	3.点击“提交”按钮，待分析的裁判文书上传服务器进行处理并返回分析结果页面
预期结果	正常解析文件内容，返回正确的分析结果

在裁判文书分析测试阶段，共计选取了50个案例（如图 4.17）作为输入进行测试，系统均能给出相应的分析结果。经过人工审校，其中共计43个案例能给出完整正确的分析结果，7个案例出现了字段缺失或错误（案例表述存在各异性，未能正确提取）的情况。分析准确率为86%,基本达到预期效果。

图 4.17: 测试案例列表截图

以下将展示裁判文书分析测试中的样例：

上传的裁判文书题为“蔡某犯故意杀人罪刑罚变更刑事裁定书”，描述的事实是故意杀人罪犯蔡某因为表现良好获得减刑，由死刑减为无期徒刑。对这裁判文书的分析结果如图 4.18所示。

通过样例可以看出，系统不仅分析出了裁判文书的各个要素，还将裁判文书中死刑剥夺政治权利，改造获得减刑的情节以“剥夺”、“改造”、“减为”等关键词的形式予以展示，满足预定需求。

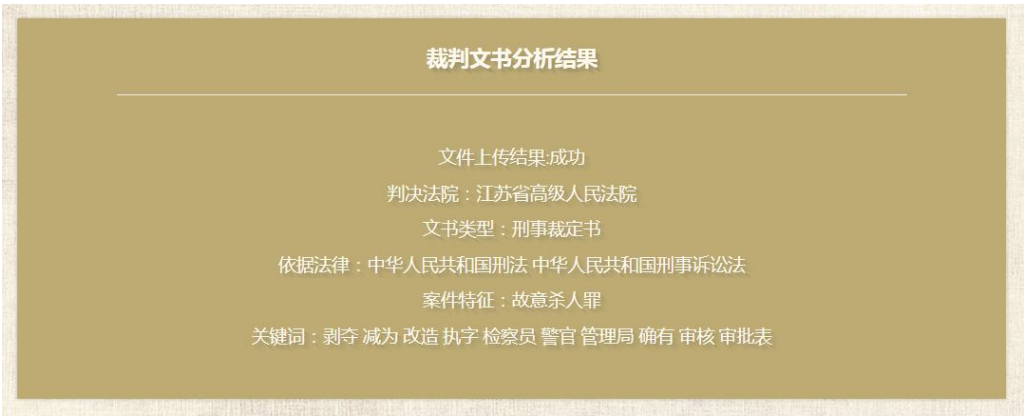

图 4.18: 裁判文书分析结果图

3. 案例筛选筛选测试

在案例筛选功能测试中，上传已知类别的案例文本，考察系统是否能够筛选数量符合预期（筛选案例的数量在上文中设定为不多于7个），并且案例内容相似（案由相同、刑期相近，文本相似度高）的文本。案例筛选功能具体测试用例如下表 4.9所示：

表 4.9: 案例筛选功能测试用例表

测试用例编号	TestCase3
测试内容	用户筛选与输入文本相似的案例
测试功能	案例筛选
测试步骤	1.访问web应用中案例筛选对应的url
	2.选定上传文件或输入文本
	3.点击“提交”按钮，案例筛选素材文本上传服务器进行处理并返回筛选结果页面
预期结果	显示输入文本内容，返回相似的筛选案例列表

在案例筛选测试阶段，共计选取了50段已明确类别的案例文本作为输入进行测试，系统均能给出相应的分析结果。

经过人工审校，其中共计37段案例文本能返回同类别的筛选结果，14段案例文本给出了相关类别的筛选结果（例如暴力抢劫行为由抢劫分类归类至故意伤害），9段案例文本出现了筛选出的案例结果与原分类无关的情形（案例文本特征较复杂，未能正确分类）。筛选结果相关比例为81%，基本达到预期效果。以下将展示案例筛选测试中的样例：

输入的待推荐文本描述了被告人与被害人因琐事纠纷而相互殴打，导致被害人轻伤二级的案件事实。此文本属于的分类为故意伤害题材，是较为典型的故意伤害事件。

系统经文本分析后，筛选出多个故意伤害类别的案例如图 4.19所示：

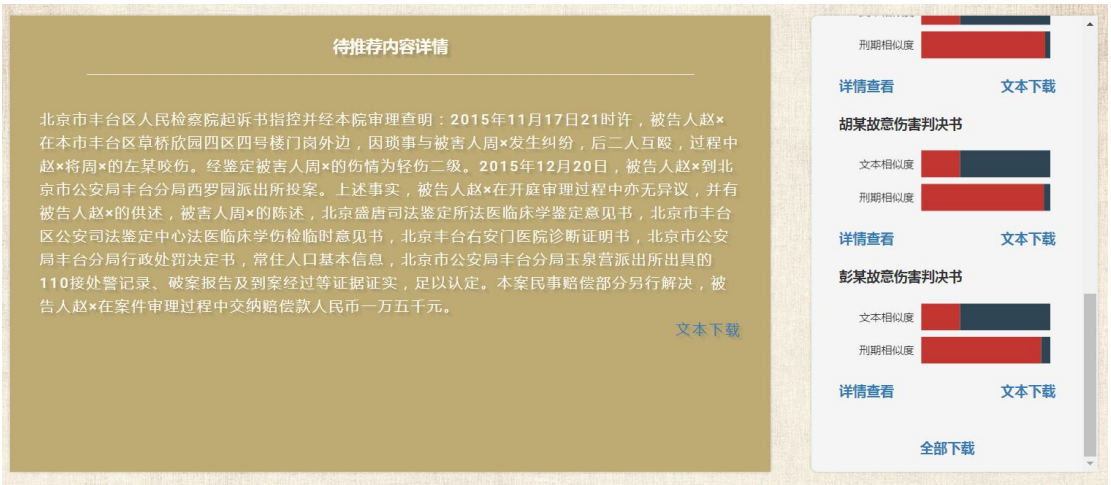

图 4.19: 案例筛选结果图

通过样例可以看出，系统能够相对准确的对输入案例文本进行类别定位，将相似案例予以展示。筛选出相似案例在数量上和内容的相似度上，均满足预定需求，完成了对相似案例的筛选工作。

4. 辅助功能测试

案例详情查看功能，要求系统能获取案例详情并予以展示，展示的结果包含案例内容和对应的推荐结果，详情查看功能具体测试用例如下表 4.10所示：

表 4.10: 详情查看功能测试用例表

测试用例编号	TestCase4
测试内容	用户查看筛选案例的详情
测试功能	详情查看
测试步骤	1.进行案例筛选操作
	2.点击“详情查看”链接
	3. 服务器进行处理并返回处理结果页面
预期结果	显示选中案例详情，更新案例筛选结果

文件下载功能包括单一文件下载和批量文件下载，以功能稍复杂的批量案例下载测试为例进行展示，其具体测试用例如下表 4.11所示：

表 4.11: 批量文件下载测试用例表

测试用例编号	TestCase5
测试内容	用户批量下载筛选案例
测试功能	批量下载
测试步骤	1.进行案例筛选操作
	2.点击“全部下载”链接
	3. 服务器进行处理并返回下载页面面
预期结果	下载含列表中全部备选案例的文本文件

经测试，系统各项辅助功能均可正常运行，满足预定需求。

4.5 本章小结

本章主要描述了各个模块的具体设计和实现。本章各节根据各模块特点对各模块的具体实现展开论述。在文本输入和文件上传模块给出了前后端的交互逻辑，选择了合适的表单提交方式。在裁判文书分析模块部分着重论述了关键信息提取和关键词提取两个子模块不同的提取方法。在案例筛选模块中，叙述了采用fastText框架进行模型训练和预测的具体实现要点，并给出筛选排序及结果展示的具体实现。最后对系统的主要功能进行了测试。

第五章　总结与展望

5.1　总结

为了帮助司法从业者进行快速地案例检索，同时也相应广大民众法律咨询的需求，本文研究了司法案例筛选系统。这是自然语言处理技术在司法领域的一次落地尝试。本文首先介绍了司法案例筛选系统所属司法智能领域的相关动态和研究现状，进一步介绍了分词、TF-IDF文本关键词提取、文本分类技术fastText和web应用框架Django等相关技术。

随后，本文详细分析了系统的功能需求和非功能需求，并生成相应的用例图和用例描述。进一步完成了系统的总体设计，明确了系统由文件上传模块、文本输入模块、裁判文书分析模块和案例筛选模块四部分模块组成。

其中，案例筛选模块是系统的核心部分，在这一模块中将实现司法案例筛选的主要功能；裁判文书分析模块是对上传裁判文书进行分析，帮助用户快速理解裁判文书；文件上传模块和文本输入模块用于和用户的交互，为整个系统的获取需要分析的文本，提供基础的支撑服务。

最后，本文介绍了各个模块的具体设计和实现要点。在文本输入和文件上传模块部分，论述了前后端的交互逻辑的处理及合适表单提交方式的选择。在裁判文书分析模块部分，依次叙述了关键信息提取和关键词提取两个子模块不同的实现方法，并交代了裁判文书及相关分析结果的存储方式。在案例筛选模块部分，叙述了采用fastText框架进行筛选模型训练及调用的具体过程，完善并实现了对应的筛选排序算法和结果展示方法。

传统筛选系统以关键字匹配的方法为主，能够处理的文本长度较短，灵活性较差，仅适合部分简单场景。本文所述的司法案例筛选系统，将目前主流的自然语言处理技术与司法场景进行有机结合，实现了对输入案情描述文字或裁判文书文本的相似案例筛选。这一系统与传统型的系统相比，能够进一步处理的篇章级的输入文本，灵活性更强，适用场景进一步得到扩展，将更有效地帮助系统使用者对获取文书的解析及筛选相似的案例。

5.2　进一步工作展望

本文初步实现了司法案例筛系统，实现了案例筛选、裁判文书分析等基本功能。在本文基础上，还有以下方面可以进一步完善：

一方面，用于模型训练的数据质量还有待提升。首先，裁判文书的数字化集中上传机制仅设立数年，这意味着时间跨度丰富的裁判文书资源仍然相对匮乏，数据缺乏一定的历史储备。其次，上传文书也存在着很明显的地域不均衡的情况。对于司法公开数字化建设较好的省份，公开的比例高，上传的数量多，而对于某些落后的省份则公开的比例较低，使得目前的可获取的裁判文书在地域维度上同样不均衡。因此，目前司法案例筛选系统依托的训练数据在时间空间上都有一定的缺陷。通过收集全面的案例数据，提高训练数据质量，能够进一步提升案例筛选的精度，使得系统的实用性进一步提升。

另一方面，与用户的交互方式由待扩展。目前系统的设计仍然停留在文本交互层面。用户使用案例筛选系统是一个复杂的场景。通过进一步的设计拓展与用户的交互方式，并从用户的使用行为中挖掘用户的使用偏好，进而推出更高效的筛选服务。甚至实现根据用户使用习惯，主动推送相关的参考案例。

司法案例筛选系统的不断进步，最终的目的不是取代人类法官进行预测和判决，而是通过技术手段简化目前繁琐的案例检索，法条研读的流程，能够使得法官、律师等司法从业者能够将大量的时间和精力用来面对需要人类主观能动性面对的问题之上。本文所述系统若能从上述两方面进行迭代与改进，将能够为司法工作者带来更优的使用体验。

参考文献

[1] 刘敏, 论司法公开的深化, 政法论丛 (2015 年06) (2015) 138–143.

[2] 张宝, 环境侵权责任构成的适用争议及其消解——基于4328 份裁判文书的实证分析, 湘潭大学学报(哲学社会科学版) (2) (2018) 10.

[3] 国务院, 国务院关于印发新一代人工智能发展规划的通知(国发[2017] 35 号) (2017).

[4] 李本, 美国司法实践中的人工智能: 问题与挑战, 中国法律评论 (2) (2018) 9.

[5] 李飞, 人工智能与司法的裁判及解释, 法律科学(西北政法大学学报) 36 (5) (2018) 32–41.

[6] Z. Qin, T. He, H. Lian, Y. Tian, J. Liu, Research on judicial data standard, in: 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), IEEE, 2018, pp. 175–177.

[7] H. Lian, T. He, Z. Qin, H. Li, J. Liu, Research on the information quality measurement of judicial documents, in: 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), IEEE, 2018, pp. 178–181.

[8] H. Lian, Z. Qin, T. He, B. Luo, Knowledge graph construction based on judicial data with social media, in: 2017 14th Web Information Systems and Applications Conference (WISA), IEEE, 2017, pp. 225–227.

[9] G. Lame, Using nlp techniques to identify legal ontology components: Concepts and relations, Artificial Intelligence and Law 12 (4) (2004) 379–396.

[10] M. R. Talib, M. K. Hanif, Z. Nabi, M. U. Sarwar, N. Ayub, Text mining of judicial system’ s corpora via clause elements, International Journal on Information Technologies & Security 9 (3).

[11] T.-K. He, H. Lian, Z.-M. Qin, Z.-Y. Chen, B. Luo, Ptm: A topic model for the inferring of the penalty, Journal of Computer Science and Technology 33 (4) (2018) 756–767.

[12] Z. Liu, H. Chen, A predictive performance comparison of machine learning models for judicial cases, in: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2017, pp. 1–6.

[13] E. Ash, D. Chen, Predicting punitiveness from judicial corpora.

[14] M. A. Neary, S. X. Chen, Artificial intelligence: legal research and law librarians, AALL Spectrum 21 (5) (2017) 16.

[15] 左卫民, 关于法律人工智能在中国运用前景的若干思考, 清华法学12 (02) (2018) 108–124.

[16] D. Said, N. M. Wanas, N. M. Darwish, N. Hegazy, A study of text preprocessing tools for arabic text categorization, in: The second international conference on Arabic language, 2009, pp. 230–236.

[17] W. J. Teahan, Y. Wen, R. McNab, I. H. Witten, A compression-based algorithm for chinese word segmentation, Computational Linguistics 26 (3) (2000) 375–393.

[18] 黄昌宁，赵海, 中文分词十年回顾, 中文信息学报 (03) (2007) 8–19.

[19] S. R. Eddy, Profile hidden markov models., Bioinformatics (Oxford, England) 14 (9) (1998) 755–763.

[20] L. R. Rabiner, B.-H. Juang, An introduction to hidden markov models, ieee assp magazine 3 (1) (1986) 4–16.

[21] C. J. Leggetter, P. C. Woodland, Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models, Computer speech & language 9 (2) (1995) 171–185.

[22] J. Ramos, et al., Using tf-idf to determine word relevance in document queries, in: Proceedings of the first instructional conference on machine learning, Vol. 242, 2003, pp. 133–142.

[23] T. Joachims, A probabilistic analysis of the rocchio algorithm with tfidf for text categorization., Tech. rep., Carnegie-mellon univ pittsburgh pa dept of computer science (1996).

[24] E.-H. S. Han, G. Karypis, Centroid-based document classification: Analysis and experimental results (2000) 424–431.

[25] R. Mihalcea, C. Corley, C. Strapparava, et al., Corpus-based and knowledge-based measures of text semantic similarity, in: AAAI, Vol. 6, 2006, pp. 775–780.

[26] H. V. Nguyen, L. Bai, Cosine similarity metric learning for face verification, in: Asian conference on computer vision, Springer, 2010, pp. 709–720.

[27] J. C. Gower, P. Legendre, Metric and euclidean properties of dissimilarity coefficients, Journal of classification 3 (1) (1986) 5–48.

[28] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM transactions on intelligent systems and technology (TIST) 2 (3) (2011) 27.

[29] G. Cai, B. Xia, Convolutional neural networks for multimedia sentiment analysis, in: Natural Language Processing and Chinese Computing, Springer, 2015, pp. 159–167.

[30] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cireşan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, L. M. Gambardella, Max-pooling convolutional neural networks for vision-based hand gesture recognition, in: 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), IEEE, 2011, pp. 342–347.

[31] A. J. Piotr Bojanowski, Edouard Grave, T. Mikolov, facebook research:fasttext, https://research.fb.com/fasttext/, august 18, 2016.

[32] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, T. Mikolov, Fasttext. zip: Compressing text classification models, arXiv preprint arXiv:1612.03651.

[33] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, Bag of tricks for efficient text classification, arXiv preprint arXiv:1607.01759.

[34] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781.

[35] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, J. C. Lai, Class-based n-gram models of natural language, Computational linguistics 18 (4) (1992) 467–479.

[36] J. Forcier, P. Bissex, W. J. Chun, Python web development with Django, Addison-Wesley Professional, 2008.

[37] A. Holovaty, J. Kaplan-Moss, The definitive guide to Django: Web development done right, Apress, 2009.

[38] 傅郁林, 民事裁判文书的功能与风格, 中国社会科学4 (128) (2000) 51.

[39] 李友根, 裁判文书公开与当事人隐私权保护, 法学5 (2010) 128–136.

[40] A. Czeskis, A. Moshchuk, T. Kohno, H. J. Wang, Lightweight server support for browser-based csrf protection, in: Proceedings of the 22nd international conference on World Wide Web, ACM, 2013, pp. 273–284.

[41] 罗东川，黄建中, 《民事案件案由规定》的理解与适用, 人民司法 (05) (2008) 18–23.

[42] 李适时, 关于《中华人民共和国刑法修正案(七)(草案)》的说明——2008 年8月25 日在第十一届全国人民代表大会常务委员会第四次会议上, 中华人民共和国全国人民代表大会常务委员会公报 (2) (2009) 190–193.

[43] J. H. Paik, A novel tf-idf weighting scheme for effective ranking, in: Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, ACM, 2013, pp. 343–352.

[44] P. Tonella, F. Ricca, E. Pianta, C. Girardi, Using keyword extraction for web site clustering (2003) 41–48.

[45] R. Rivest, The md5 message-digest algorithm, Tech. rep. (1992).

[46] C. Xiao, H. Zhong, Z. Guo, C. Tu, Z. Liu, M. Sun, Y. Feng, X. Han, Z. Hu, H. Wang, et al., Cail2018: A large-scale legal dataset for judgment prediction, arXiv preprint arXiv:1807.02478.

[47] 伍红梅, 以“大数据+机器学习”为驱动构建刑事案件判案智能预测系统, 人民司法(应用) 10 (2018) 34–40.

[48] 张保生, 人工智能法律系统的法理学思考, 法学评论5 (2001) 11–21.

[49] G. E. Hinton, R. R. Salakhutdinov, Replicated softmax: an undirected topic model, in: Advances in neural information processing systems, 2009, pp. 1607–1614.

[50] P.-T. De Boer, D. P. Kroese, S. Mannor, R. Y. Rubinstein, A tutorial on the cross-entropy method, Annals of operations research 134 (1) (2005) 19–67.

[51] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, L. D. Jackel, Handwritten digit recognition with a back-propagation network, in: Advances in neural information processing systems, 1990, pp. 396–404.

[52] G. A. Miller, The magical number seven, plus or minus two: Some limits on our capacity for processing information., Psychological review 63 (2) (1956) 81.

[53] R. W. Proctor, D. W. Schneider, Hick' s law for choice reaction time: A review, The Quarterly Journal of Experimental Psychology (just-accepted) (2017) 1–56.

[54] D. Li, H. Mei, Y. Shen, S. Su, W. Zhang, J. Wang, M. Zu, W. Chen, Echarts: A declarative framework for rapid construction of web-based visualization, Visual Informatics 2 (2) (2018) 136–146.

简历与科研成果

基本情况 秦泽民，男，汉族，1995 年 6 月出生，江苏省无锡市人。

教育背景

2017.9～2019.6 南京大学软件学院 硕士

2013.9～2017.7 大连海事大学信息科学技术学院 本科

读研期间的成果（包括发表的论文及参与的专利）：

1. **Qin Z**, Lian H, He T, et al. Cluster Correction on Polysemy and Synonymy[C]//2017 14th Web Information Systems and Applications Conference (WISA). IEEE, 2017: 136-138.

2. **Qin Z**, He T, Lian H, et al. Research on Judicial Data Standard[C]//2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 2018: 175-177.

3. Lian H, **Qin Z**, He T, et al. Knowledge Graph Construction Based on Judicial Data with Social Media[C]//2017 14th Web Information Systems and Applications Conference (WISA). IEEE, 2017: 225-227.

4. He T K, Lian H, **Qin Z M**, et al. PTM: A Topic Model for the Inferring of the Penalty[J]. Journal of Computer Science and Technology, 2018, 33(4): 756-767.

5. Lian H, He T, **Qin Z**, et al. Research on the Information Quality Measurement of Judicial Documents[C]//2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 2018: 178-181.

6. 陈振宇、何铁科、**秦泽民**、廉昊、骆斌、李玉莹、张欣，“一种基于k-means的类案推荐方法”，申请号：201711200604.0，已受理

7. 陈振宇、何铁科、廉昊、**秦泽民**、骆斌、李玉莹、张欣，“一种问答社区中的感知信息质量度量方法”，申请号：201711200605.5，已受理

8. 何铁科、严格、廉昊、**秦泽民**、史洋洋、陈振宇，“一种用于刑罚推断的主题模型PTM”，申请号：201810561189.X，已受理

9. 何铁科、**秦泽民**、严格、陈振宇、周业茂、骆斌、李玉莹，“一种司法领域数据的规范方法”，申请号：201810561376.8，已受理

致　谢

在本文完成之际，我想向所有帮助过我的老师、同学等表示衷心的感谢。

首先，感谢我的指导老师刘嘉老师和何铁科老师。这一论文在写作的过程中离开不了两位老师的谆谆教诲。在论文选题之时，两位老师与我多次探讨，最终才谨慎地确定了选题。在论文的提纲拟定、内容编排方面两位老师都给予了我非常有指导意义的建议。

其次我要感谢实验室一起工作和研究的同学们。感谢他们在我遇到技术难题时，与我一同分析调试；在我遇到自己不熟悉的技术领域时，热心地向我提供帮助和引导。

最后，我想感谢其他所有帮助过我的老师和同学。他们的建言献策帮助突破了一些研究时遇到的困难。同时，向论文评阅及论文答辩委员会的老师致以最诚挚的谢意，感谢各位老师的辛勤工作。